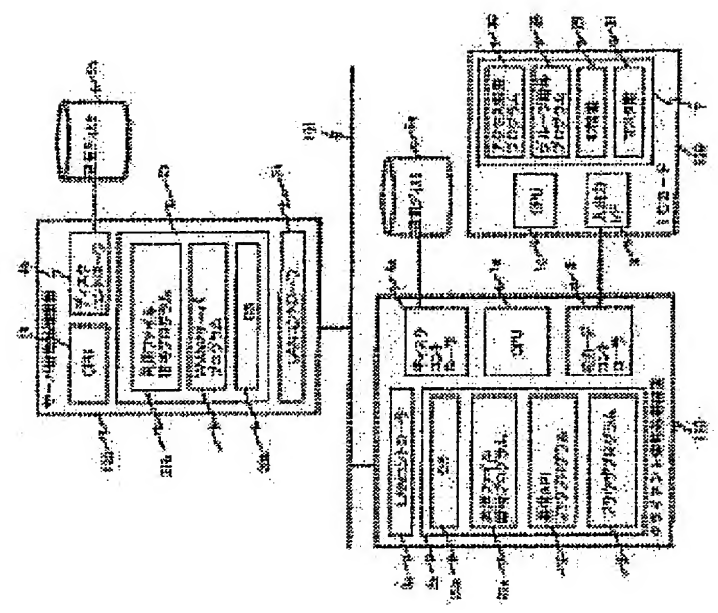


**FILE CIPHERING SYSTEM AND ITS CONTROL METHOD, AND CIPHER  
FILE RECEPTION SYSTEM AND ITS CONTROL METHOD**

Publication number: JP9251426 (A)  
Publication date: 1997-09-22  
Inventor(s): ARAI MASATO; ITO HIROMICHI; SUZAKI SEIICHI; UMEKI HISASHI; OTSU YUTAKA; MORIFUJI HAJIME; SHIMIZU MAYUKO  
Applicant(s): HITACHI LTD  
Classification:  
- international: G06F12/14; G06F12/00; G06F13/00; G06F21/24; G06F21/14; G06F12/00; G06F13/00; G06F21/00; (IPC1-7): G06F12/14; G06F12/00; G06F13/00  
- European:  
Application number: JP19960182977 19960712  
Priority number(s): JP19960182977 19960712; JP19960002046 19960110

**Abstract of JP 9251426 (A)**

**PROBLEM TO BE SOLVED:** To provide a safe cipher file reception system and its method which need only one cipher file even when plural users access the same WWW(world wide web).  
**SOLUTION:** This cipher file reception system includes a server information processor 120 which includes a server file cipher program 21b and previously ciphers the information transmitted through a WWW to store them in a magnetic disk 5b, and a client information processor 110 which includes a shared file cipher program 21a and a communication API hook program. Then the system hooks the communication API call sent from a browser program and automatically decodes the received file data.





(19) 日本国特許庁 (J P)

## (12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-251426

(43) 公開日 平成9年(1997)9月22日

(51) Int. Cl. <sup>a</sup>	識別記号	F I		
G06F 12/14	320	G06F 12/14	320	B
12/00	537	12/00	537	H
	545		545	F
13/00	353	13/00	353	M

審査請求 未請求 請求項の数39 O L (全33頁)

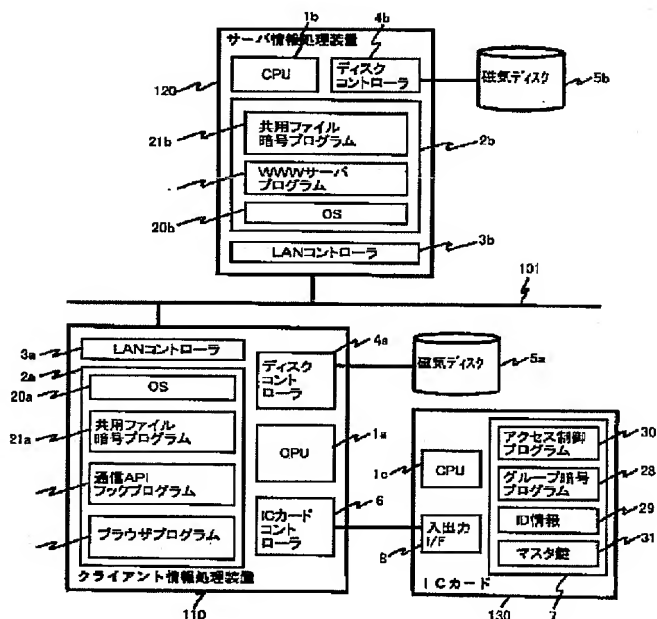
(21) 出願番号	特願平8-182977	(71) 出願人	000005108 株式会社日立製作所 東京都千代田区神田駿河台四丁目6番地
(22) 出願日	平成8年(1996)7月12日	(72) 発明者	荒井 正人 神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内
(31) 優先権主張番号	特願平8-2046	(72) 発明者	伊藤 浩道 神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内
(32) 優先日	平8(1996)1月10日	(72) 発明者	洲崎 誠一 神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内
(33) 優先権主張国	日本(J P)	(74) 代理人	弁理士 沼形 義彰 (外1名) 最終頁に続く

(54) 【発明の名称】 ファイル暗号化システム及びその制御方法ならびに暗号ファイル受信システム及びその制御方法

(57) 【要約】

【課題】 複数のユーザーが同一のファイルをWWWでアクセスする場合にも、暗号ファイルが一個で済み、且つ安全な暗号ファイル受信システム及びその制御方法を提供する。

【解決手段】 共用ファイル暗号プログラム21bを有し、WWWで発信する情報を予め暗号化して磁気ディスク5bに格納したサーバ情報処理装置120と、共用ファイル暗号プログラム21aと通信APIフックプログラム27を設けたクライアント情報処理装置110とからなり、ブラウザプログラム19からの通信API呼び出しをフックし、受信したファイルデータを自動的に復号化するようにした暗号ファイル受信システム。



## 【特許請求の範囲】

【請求項 1】 少なくともファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置あるいはネットワークで接続された第二の情報処理装置上の記憶装置へのファイル格納手段とを具備する情報処理装置からなるファイル暗号化システムにおいて、暗号化を行う対象ディレクトリを予め設定しファイルに記憶する暗号化ディレクトリ記憶手段と、前記 OS 上で動作するアプリケーションプログラムから前記ディレクトリ内のファイルへの書き込みデータを暗号化する暗号化手段とを設けたことを特徴とするファイル暗号化システム。

【請求項 2】 前記暗号化ディレクトリ記憶手段は、前記情報処理装置を利用するユーザー毎に、独立した複数の暗号化対象ディレクトリを設定可能であることを特徴とする請求項 1 記載のファイル暗号化システム。

【請求項 3】 前記暗号化ディレクトリ記憶手段は、前記対象ディレクトリ毎に復号可能なユーザーのリストを記憶することを特徴とする請求項 1 または請求項 2 記載のファイル暗号化システム。

【請求項 4】 前記ユーザーのリストを記憶したファイルを、前記対象ディレクトリ毎に前記対象ディレクトリ内に作成することを特徴とする請求項 3 記載のファイル暗号化システム。

【請求項 5】 前記暗号化手段は、当該ディレクトリに暗号化対象ディレクトリであることを示すファイルがある場合にファイルへの書き込みデータを暗号化することを特徴とする請求項 1 記載のファイル暗号化システム。

【請求項 6】 前記ユーザーのリストを暗号化し記憶することを特徴とする請求項 3 ないし請求項 5 のいずれか記載のファイル暗号化システム。

【請求項 7】 前記アプリケーションプログラムから見た論理的ファイル名と、前記暗号化手段によって格納される物理的ファイル名とが異なることを特徴とする請求項 1 ないし請求項 6 のいずれか記載のファイル暗号化システム。

【請求項 8】 少なくともファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置あるいはネットワークで接続された第二の情報処理装置上の記憶装置へのファイル格納手段とを具備する情報処理装置からなるファイル暗号化システムにおいて、暗号化を行う対象ディレクトリを予め設定しファイルに記憶する暗号化ディレクトリ記憶手段と、アプリケーションプログラムから前記ディレクトリ内のファイルへの書き込みデータの暗号化と読み出しデータの復号化とを行う暗号復号手段とを設けたことを特徴とするファイル暗号化システム。

【請求項 9】 前記暗号復号手段は、アプリケーションプログラムからのファイル書き込み或いは読み出しのサイズとは独立したサイズの平文ブロックバッファと暗号

文ブロックバッファとを具備し、暗号化および復号化の処理は、前記ブロックバッファ上の全データを一度に処理することによって行うことを特徴とする請求項 8 記載のファイル暗号化システム。

【請求項 10】 前記暗号化手段あるいは復号化手段は、前記アプリケーションプログラムから前記 OS に発行されるファイル操作要求をフックするフック手段によって呼び出されることを特徴とする請求項 1 ないし請求項 9 のいずれか記載のファイル暗号化システム。

【請求項 11】 前記暗号化手段は、復号を許可するユーザーのリストの文字列をハッシュ関数によって圧縮することによって作成した数値を暗号化の鍵として用いることを特徴とする請求項 1 ないし請求項 10 のいずれか記載のファイル暗号化システム。

【請求項 12】 前記暗号化手段は、復号を許可するユーザーのリストの文字列をハッシュ関数によって圧縮することによって作成した数値で暗号化の鍵を暗号化し、該暗号化鍵を暗号化ファイルに付加することを特徴とする請求項 1 ないし請求項 10 のいずれか記載のファイル暗号化システム。

【請求項 13】 少なくともファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置あるいはネットワークで接続された第二の情報処理装置上の記憶装置へのファイル格納手段と、前記ファイルを暗号化するファイル暗号化手段とを具備する情報処理装置からなるファイル暗号化システムの制御方法において、前記ファイル暗号化手段が、アプリケーションプログラムが解釈し処理する文法に従って記述された平文ファイルのデータ暗号化に際し、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子を前記暗号化後のデータに付加し格納することを特徴とするファイル暗号化システムの制御方法。

【請求項 14】 前記暗号化後のデータに前記注釈文の終了を示す記述子が存在しないような暗号化鍵を選択し使用することを特徴とする請求項 13 記載のファイル暗号化システムの制御方法。

【請求項 15】 ファイルが暗号文のファイルであることを前記アプリケーションプログラムが表示するための記述文を、前記暗号化後のデータを含む前記注釈文の前あるいは後ろに付加し格納することを特徴とする請求項 13 又は請求項 14 記載のファイル暗号化システムの制御方法。

【請求項 16】 少なくともファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置あるいはネットワークで接続された第二の情報処理装置上の記憶装置へのファイル格納手段と、前記記憶装置に格納されたファイルを復号化するファイル復号化手段とを具備する情報処理装置からなるファイル暗号化システムの制御方法において、前記

ファイル復号化手段が、アプリケーションプログラムが解釈し処理する文法に従って記述された暗号ファイルのデータ復号化に際し、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子によって注釈文とされたデータ部分を復号化し、該復号化後のデータを暗号化前の平文ファイルのデータとして格納することを特徴とするファイル暗号化システムの制御方法。

【請求項 17】 前記暗号化手段あるいは復号化手段は、前記アプリケーションプログラムから前記 OS に発行されるファイル操作要求をフックするフック手段によって呼び出されることを特徴とする請求項 13 ないし請求項 16 のいずれか記載のファイル暗号化システムの制御方法。

【請求項 18】 前記暗号化手段は、復号を許可するユーザーのリストの文字列をハッシュ関数によって圧縮することによって作成した数値を暗号化の鍵として用いることを特徴とする請求項 13 ないし請求項 17 のいずれか記載のファイル暗号化システムの制御方法。

【請求項 19】 前記暗号化手段は、復号を許可するユーザーのリストの文字列をハッシュ関数によって圧縮することによって作成した数値で暗号化の鍵を暗号化し、該暗号化鍵を暗号化ファイルに付加することを特徴とする請求項 13 ないし請求項 17 のいずれか記載のファイル暗号化システムの制御方法。

【請求項 20】 少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置とはネットワークで接続され、少なくとも通信制御機能を持つオペレーティングシステム（OS）と、該 OS 上で動作し、前記第一の情報処理装置上の記憶装置からファイルを受信する機能を持つアプリケーションプログラムとを具備する第二の情報処理装置によって構成された暗号ファイル受信システムにおいて、前記第一の情報処理装置が、前記ファイルを暗号化するファイル暗号化手段を有し、前記第二の情報処理装置が、前記アプリケーションプログラムが前記第一の情報処理装置からファイル暗号化手段によって暗号化された前記暗号ファイルを受信する際に、該受信ファイルを復号化してから前記アプリケーションプログラムに渡す受信ファイル復号化手段を有することを特徴とする暗号ファイル受信システム。

【請求項 21】 受信ファイル復号化手段は、前記ファイル暗号化手段によって前記暗号ファイルに復号可能なユーザーのリストが付加されたファイルを受信して、復号化することを特徴とする請求項 20 記載の暗号ファイル受信システム。

【請求項 22】 前記受信ファイル復号化手段は、アプリケーションプログラムが前記 OS に発行するデータ送受信要求をフックするフック手段によって呼び出される

ことを特徴とする請求項 20 又は請求項 21 記載の暗号ファイル受信システム。

【請求項 23】 前記フック手段は、前記第二の情報処理装置のユーザーの身元を確認するユーザー認証手段によって認証されたユーザーが存在する場合にのみ前記受信ファイル復号化手段を呼び出すことを特徴とする請求項 20 ないし請求項 22 のいずれか記載の暗号ファイル受信システム。

【請求項 24】 前記受信ファイル復号化手段は、前記ユーザー認証手段によって認証されたユーザーが、前記復号可能なユーザーのリストに含まれているか否かを判別する復号権利判別手段によって復号権利があると判断された場合にのみ呼び出されることを特徴とする請求項 20 ないし請求項 23 のいずれか記載の暗号ファイル受信システム。

【請求項 25】 前記暗号ファイルに、復号可能なユーザーのリストを暗号化して付加したファイルを受信して復号化することを特徴とする請求項 20 記載の暗号ファイル受信システム。

【請求項 26】 前記第一の情報処理装置上の記憶装置には、前記ファイル暗号化手段によって予め暗号化されたファイルが格納されていることを特徴とする請求項 20 ないし請求項 25 のいずれか記載の暗号ファイル受信システム。

【請求項 27】 前記フック手段は、前記アプリケーションプログラムが、前記第一の情報処理装置へ送信したファイル転送要求コマンドに対する受信ファイルのみを復号化の対象として処理することを特徴とする請求項 20 ないし請求項 25 のいずれか記載の暗号ファイル受信システム。

【請求項 28】 少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを暗号化するファイル暗号化手段と、前記暗号化ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置とはネットワークで接続され、少なくとも通信制御機能を持つオペレーティングシステム（OS）と、該 OS 上で動作し、前記第一の情報処理装置上の記憶装置から前記暗号ファイルを受信する機能を持つアプリケーションプログラムとを具備する第二の情報処理装置から構成された暗号ファイル受信システムにおいて、前記ファイル暗号化手段は、前記アプリケーションプログラムが解釈し処理する文法に従って記述された平文ファイルのデータ暗号化において、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子を、前記暗号化後のデータに付加したものを暗号ファイルとするものであり、前記アプリケーションプログラムが前記暗号ファイルを受信したとき、前記記述子が付加されて注釈文となっているデータ部分のみを取り出して復号化する受信ファイル復号化手段を設

けたことを特徴とする暗号ファイル受信システム。

【請求項 2 9】 少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを暗号化するファイル暗号化手段と、前記暗号化ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置とはネットワークで接続され、少なくとも通信制御機能を持つオペレーティングシステム（OS）を具備する第二の情報処理装置と、前記ネットワークで接続され、前記第二の情報処理装置が前記第一の情報処理装置へ送信するデータや、反対に前記第二の情報処理装置が前記第一の情報処理装置から受信するデータを中継する機能と、該受信データをキャッシュするキャッシュ機能とを具備する第三の情報処理装置とから構成される暗号化ファイル受信システムにおいて、前記第二の情報処理装置が、前記暗号ファイルの転送要求を出すと、該暗号ファイルが前記第一の情報処理装置から前記第三の情報処理装置を経由して前記第二の情報処理装置へ転送されると共に、該暗号ファイルは前記キャッシュ機能によってキャッシュされ、前記第二の情報処理装置が、再度同じ暗号ファイルの転送要求を出すと、前記キャッシュ機能によりキャッシュされた暗号ファイルが前記第三の情報処理装置から前記第二の情報処理装置へ転送されることを特徴とする暗号ファイル受信システム。

【請求項 3 0】 少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置とはネットワークで接続され、少なくとも通信制御機能を持つオペレーティングシステム（OS）と、該 OS 上で動作し、前記第一の情報処理装置上の記憶装置からファイルを受信する機能を持つアプリケーションプログラムとを具備する第二の情報処理装置によって構成された暗号ファイル受信システムの制御方法において、前記アプリケーションプログラムが前記第一の情報処理装置から前記暗号ファイルを受信する際に、該受信ファイルを復号化してから前記アプリケーションプログラムに渡すようにしたことを特徴とする暗号ファイル受信システムの制御方法。

【請求項 3 1】 前記ファイル暗号化手段によって、前記暗号ファイルに復号可能なユーザーのリストが付加されたファイルを受信して、復号化するようにしたことを特徴とする請求項 3 0 記載の暗号ファイル受信システムの制御方法。

【請求項 3 2】 前記受信ファイル復号化手段を、アプリケーションプログラムが前記 OS に発行するデータ送受信要求をフックするフック手段によって呼び出すことを特徴とする請求項 3 0 記載の暗号ファイル受信システムの制御方法。

【請求項 3 3】 前記フック手段は、前記第二の情報処

理装置のユーザーの身元を確認するユーザー認証手段によって認証されたユーザーが存在する場合にのみ前記受信ファイル復号化手段を呼び出すことを特徴とする請求項 3 0 ないし請求項 3 2 のいずれか記載の暗号ファイル受信システムの制御方法。

【請求項 3 4】 前記受信ファイル復号化手段を、前記ユーザー認証手段によって認証されたユーザーが、前記復号可能なユーザーのリストに含まれているか否かを判別する復号権利判別手段によって復号権利があると判断された場合にのみ呼び出すことを特徴とする請求項 3 0 ないし請求項 3 3 のいずれか記載の暗号ファイル受信システムの制御方法。

【請求項 3 5】 前記暗号ファイルに、復号可能なユーザーのリストを暗号化して付加したファイルを受信して復号化することを特徴とする請求項 3 0 記載の暗号ファイル受信システムの制御方法。

【請求項 3 6】 前記第一の情報処理装置上の記憶装置に、前記ファイル暗号化手段によって予め暗号化されたファイルを格納することを特徴とする請求項 3 0 ないし請求項 3 5 のいずれか記載の暗号ファイル受信システムの制御方法。

【請求項 3 7】 前記フック手段は、前記アプリケーションプログラムが、前記第一の情報処理装置へ送信したファイル転送要求コマンドに対する受信ファイルのみを復号化の対象として処理することを特徴とする請求項 3 0 ないし請求項 3 5 記載の暗号ファイル受信システムの制御方法。

【請求項 3 8】 少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを暗号化するファイル暗号化手段と、前記暗号化ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置とはネットワークで接続され、少なくとも通信制御機能を持つオペレーティングシステム（OS）と、該 OS 上で動作し、前記第一の情報処理装置上の記憶装置から前記暗号ファイルを受信する機能を持つアプリケーションプログラムとを具備する第二の情報処理装置から構成された暗号ファイル受信システムの制御方法において、前記ファイル暗号化手段は、前記アプリケーションプログラムが解釈し処理する文法に従って記述された平文ファイルのデータ暗号化において、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子を、前記暗号化後のデータに付加したものを暗号ファイルとし、前記アプリケーションプログラムが前記暗号ファイルを受信したとき、前記記述子が付加されて注釈文となっているデータ部分のみを取り出して復号化することを特徴とする暗号ファイル受信システムの制御方法。

【請求項 3 9】 少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム（O

10

20

30

40

50

S) と、前記ファイルを暗号化するファイル暗号化手段と、前記暗号化ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置とはネットワークで接続され、少なくとも通信制御機能を持つオペレーティングシステム (OS) を具備する第二の情報処理装置と、前記ネットワークで接続され、前記第二の情報処理装置が前記第一の情報処理装置へ送信するデータや、反対に前記第二の情報処理装置が前記第一の情報処理装置から受信するデータを中継する機能と、該受信データをキャッシュするキャッシュ機能とを具備する第三の情報処理装置とから構成される暗号ファイル受信システムの制御装置において、前記第二の情報処理装置が前記暗号ファイルの転送要求を出すと、該暗号ファイルが前記第一の情報処理装置から前記第三の情報処理装置を経由して前記第二の情報処理装置へ転送されると共に、該暗号ファイルは前記キャッシュ機能によってキャッシュされ、前記第二の情報処理装置が、再度同じ暗号ファイルの転送要求を出すと、前記キャッシュ機能によりキャッシュされた暗号ファイルを前記第三の情報処理装置から前記第二の情報処理装置へ転送することを特徴とする暗号ファイル受信システムの制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、情報処理装置のファイルの暗号化システムおよびその制御方法ならびに通信回線を通じて暗号ファイルを受信する際のデータ受信制御方法に係り、特に、複数のユーザーが復号可能な暗号ファイルを作成あるいは復号化する場合に好適な、共用ファイル暗号システム及びその制御方法ならびにアプリケーションプログラムにとって透過な暗号ファイル受信システム及びその制御方法に関する。

【0002】

【従来の技術】パーソナルコンピュータなどの情報処理装置が普及するにつれて、各情報処理装置毎の磁気記憶装置上に保管していたユーザーのプログラムファイル、データファイルなどを一箇所の大容量磁気記憶装置に保管し、各情報処理装置のユーザーがファイルを共用する、といった使い方が行われるようになってきた。一般的には、ファイルサーバプログラムが搭載されたサーバ情報処理装置と該サーバ情報処理装置へアクセスするためのクライアントプログラムが搭載された複数のクライアント情報処理装置をネットワークで接続し、サーバ情報処理装置上のファイルを各クライアント情報処理装置からアクセスし利用する。このようなファイル共用システムは、従来は企業等の内部で閉じていたが、インターネットと呼ばれる世界的規模のネットワークの普及と共に、このインターネットと企業内ネットワークを接続し企業外からもアクセス可能とするケースが増大している。上述のようなファイル共用システムの普及とネットワークの広域化は、ファイルサーバ上の共用ファイルの

セキュリティ即ち不正アクセスに対する防御の必要性を従来にも増して高くしている。

【0003】サーバ情報処理装置上のファイルのセキュリティを保つ方法としては、ファイルサーバプログラムが具備するユーザー認証機構と、該ユーザー認証に基づいた、各ファイルへのアクセス権限設定/確認機構を用いるのが一般的である。しかし、この方式ではネットワーク上を流れるデータの暗号化が行われなため、盗聴による不正アクセスが可能である。そこで、より高いセキュリティが必要な場合には、格納しているファイル或いはネットワーク上を流れるファイルのデータを秘密鍵や公開鍵を用いて暗号化するのが一般的である。これらの暗号化方法については、「データ保護と暗号化の研究」一松信監修、日本経済新聞社 (昭58) などで詳しく開示されている。

【0004】また、前記インターネットにおいては、情報発信機能を持つWWW (World-Wide Web) サーバを利用し、ハイパーテキストと呼ばれる形式の文書ファイルで情報を提供するのが一般的となっている。前記ハイパーテキストを解釈し、グラフィカルに表示するソフトウェアはブラウザと呼ばれ、広く普及している。前記ハイパーテキストの言語仕様としては、HTML (HyperText Markup Language) が一般に用いられており、この言語仕様やHTMLで記述されたファイルを表示するブラウザについては、「UNIX MAGAZINE」(1994. 10) pp. 52-60などで詳しく述べられている。なお、殆どのHTMLファイルは他のHTMLファイルやイメージデータファイルとリンクしており、前記ブラウザはHTMLファイルを受信した後で、それがリンクしているイメージデータファイルを受信し、該HTMLファイルの内容と合わせて同一画面に表示する。これらHTMLファイルとそれにリンクされたイメージデータファイルを総括してページと呼ぶ。

【0005】上記WWWサーバは、HTTP (Hypertext Transfer Protocol) という通信規約に基づいて、上記HTMLファイルやイメージデータをブラウザに送信する。この時、インターネット上を流れるデータは暗号化されていないため、WWWサーバに機密情報を含むページを格納することはセキュリティ上問題がある。インターネット上を流れるデータの暗号化手段としては、例えばSecureHTTPやSSL (Secure Socket Layer) などが挙げられる。また、これらの技術については、「日経コミュニケーション」日経BP社 (1995. 12. 4) p75-80) などに記載されている。

【0006】

【発明が解決しようとする課題】上述の暗号化は不正アクセスに対する防御ならびにデータの盗聴や改ざんに対する防御という観点からは十分ではあるが、複数のユー



ザーが同一のファイルを安全にアクセスできるようにするために、ユーザー毎に異なる鍵を用いる必要があり鍵の管理が困難であるという課題がある。また、格納されているファイルを暗号化する場合には、鍵毎に暗号化ファイルを作成する必要がある、記憶装置の記憶容量を多く消費してしまうという課題がある。また、通信時にネットワーク上のデータを暗号化する方法においては、暗号/復号処理のためアクセス速度が低下してしまうという課題もある。さらに、通信時にネットワーク上のデータを暗号化する方法においては、暗号化処理のためアクセス速度が低下してしまうという課題と、WWWサーバが機密情報を平文ファイルで管理しているため、該平文ファイルを管理する情報処理装置に直接アクセスすることで機密情報が漏れてしまうという課題とがある。

【0007】また、ファイル共用システムの場合、クライアント情報処理装置が必ずしも復号化ソフトウェアプログラムを具備しているとは限らない。暗号化されたファイルを復号化ソフトウェアが具備されていない情報処理装置上のアプリケーションプログラムで読み出した場合、該情報処理装置のモニタ画面上に意味のない文字や記号が表示されたり、該アプリケーションプログラムが異常動作することがあるという課題がある。

【0008】また、格納するファイルを暗号化する方法では、ユーザーが暗号化操作を忘れてしまい、不正なアクセスから防御すべきファイルを平文で格納してしまうという課題がある。

【0009】また、インターネット上で上記ブラウザを具備するクライアント情報処理装置が必ずしも復号化ソフトウェアプログラムを具備しているとは限らない。暗号化されたファイルを復号化ソフトウェアが具備されていない情報処理装置上のブラウザで読み出した場合、該情報処理装置のモニタ画面上に意味のない文字や記号が表示されたり、該ブラウザプログラムが異常動作することがあるという課題がある。

【0010】そこで、本発明の目的は、複数のユーザーが同一のファイルを利用する場合にも暗号化ファイルが一個で済み且つ安全なファイル暗号化システムおよびその制御方法を提供することにある。

【0011】また、本発明の他の目的は、複数のユーザーが同一のディレクトリ内のファイルを暗号化する場合に、復号化を許可するユーザーをユーザー毎に設定可能なファイル暗号化システムおよびその制御方法を提供することにある。

【0012】また本発明の他の目的は、復号化ソフトウェアを具備していないクライアント情報処理装置において暗号化されたファイルを読み出した場合にも、異常動作することのないファイル暗号化システムおよびその制御方法を提供することにある。

【0013】また、本発明の他の目的は、ユーザーが暗号化を忘れることのないファイル暗号化システムおよび

その制御方法を提供することにある。

【0014】さらに、本発明の目的は、インターネット上で複数のユーザーが同一のページをWWWサーバから受信する場合にも暗号ファイルが1ファイルにつき一個で済み、且つ安全な暗号ファイル受信システム及びその制御方法を提供することにある。

【0015】また、本発明の他の目的は、復号化ソフトウェアを具備していないクライアント情報処理装置において暗号化されたファイルを読み出した場合にも、異常動作することのない暗号ファイル受信システム及びその制御方法を提供することにある。

【0016】さらに、本発明の他の目的は、受信した暗号ファイルの復号化処理を、ユーザーがファイル毎に明示的に指示する必要のない暗号ファイル受信システム及びその制御方法を提供することにある。

【0017】

【課題を解決するための手段】上記目的は、少なくともファイルへの入出力制御機能を持つオペレーティングシステム(OS)と、前記ファイルを格納する記憶装置とを具備する情報処理装置において、暗号化を行う対象ディレクトリを予めユーザー毎に複数設定可能な暗号化ディレクトリ記憶手段と、前記記憶手段の記憶データを暗号化する手段と、前記OS上で動作するアプリケーションプログラムから前記ディレクトリ内のファイルへの書き込みデータを暗号化する暗号化手段とを設けることによって達成される。

【0018】また、上記他の目的は、少なくともファイルへの入出力制御機能を持つオペレーティングシステム(OS)と、前記ファイルを格納する記憶装置と、前記ファイルを暗号化するファイル暗号化手段と、前記ファイルを復号化する復号化手段とを具備する情報処理装置において、前記ファイル暗号化手段は、アプリケーションプログラムが解釈し処理する文法に従って記述された平文ファイルのデータ暗号化において、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子を前記暗号化後のデータに付加し暗号ファイルとして格納し、前記ファイル復号化手段は、前記暗号ファイルのデータ復号化において、前記注釈文のための記述子によって注釈文とされたデータ部分を復号化し、暗号化前の平文ファイルのデータとして格納することによって達成される。

【0019】また、上記他の目的は、前記アプリケーションプログラムから前記OSに発行されるファイル操作要求をフックするフック手段を設け、前記ファイル操作要求発行時に自動的に暗号化あるいは復号化を行う手段を設けることによって達成される。

【0020】上記目的は、少なくとも通信制御機能とファイルへの入出力制御機能を持つオペレーティングシステム(OS)と、前記ファイルを格納する記憶装置とを具備する第一の情報処理装置と、該第一の情報処理装置



とはネットワークで接続され、少なくとも前記第一の情報処理装置上の記憶装置からファイルを受信するための通信制御機能を持つオペレーティングシステム（OS）を具備する第二の情報処理装置から構成されるシステムにおいて、前記ファイルを暗号化して該暗号ファイルに情報開示先となるユーザーのリストを添付するファイル暗号化手段と、前記第二の情報処理装置を利用するユーザーが、前記記憶装置に登録された前記暗号ファイルを受信したときに、該暗号ファイルを復号化する受信ファイル復号化手段と、前記ユーザーが前記ユーザーリスト

に含まれている場合には、前記受信ファイル復号化手段を呼び出し、含まれていない場合には復号化を行わないよう制御する復号権利判別手段とを設けることによって達成される。

【0021】また、上記他の目的は、少なくともファイルへの入出力制御機能を持つオペレーティングシステム（OS）と、前記ファイルを格納する記憶装置と、前記ファイルを暗号化するファイル暗号化手段とを具備する第一の情報処理装置において、前記ファイル暗号化手段は、アプリケーションプログラムが解釈し処理する文法に従って記述された平文ファイルのデータ暗号化において、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子を前記暗号化後のデータに付加し暗号ファイルとして格納し、前記受信ファイル復号化手段は、前記暗号ファイルのデータ復号化において、前記注釈文のための記述子によって注釈文とされたデータ部分を復号化し、暗号化前の平文ファイルのデータとしてアプリケーションプログラムに渡すことによって達成される。

【0022】また、上記他の目的は、前記第二の情報処理装置において、前記アプリケーションプログラムから前記OSに発行されるデータ送受信要求をフックするフック手段を設け、該フック手段により前記アプリケーションプログラムより先に受信データをチェックし、該受信データが前記暗号ファイルのデータであれば復号化を行い、前記アプリケーションプログラムに渡すことによって達成される。

【0023】

【発明の実施の形態】以下、本発明の一実施例を図を用いて説明する。

【0024】まず、全体のシステム構成を説明する。図1は、本発明の共用ファイル暗号システムの一構成例である。110はクライアント情報処理装置、120はサーバ情報処理装置である。サーバ情報処理装置120には、中央処理装置（CPU）1b、メモリ2b、LANコントローラ3b、ディスクコントローラ4b、磁気ディスク5bが具備されている。前記サーバ情報処理装置120の起動時には、オペレーティングシステム（OS）20bおよびファイルサーバプログラム25が、ディスクコントローラ4bを介して磁気ディスク5bから

メモリ2b上にロードされる。クライアント情報処理装置110には、CPU1a、メモリ2a、LANコントローラ3a、ディスクコントローラ4a、磁気ディスク5a、ICカードコントローラ6がそれぞれ具備されている。前記クライアント情報処理装置110の起動時には、OS20a、クライアントプログラム24、共用ファイル暗号プログラム21、ファイルI/Oフックプログラム22が、前記ディスクコントローラ4aを介して前記磁気ディスク5aから前記メモリ2a上にロードされる。また、アプリケーションプログラム23は、前記クライアント情報処理装置110のユーザー（以下、単にユーザーと呼ぶ）の指示によって前記ディスクコントローラ4aを介して前記磁気ディスク5aから前記メモリ2a上にロードされる。前記ファイルI/Oフックプログラム22は、アプリケーションプログラム23が前記OS20aに発行するファイルI/O APIをフック（横取り）し、ファイルデータの暗号化・復号化をバックグラウンドで自動的に行うものである。

【0025】前記サーバ情報処理装置120と前記クライアント情報処理装置110は、ローカルエリアネットワーク（LAN）101を介して相互に接続されており、各々のLANコントローラ3を介して通信を行う。図1では一台のクライアント情報処理装置110を示したが、実際には、同じ構成のクライアント情報処理装置110が複数前記LAN101に接続されており、これらのクライアント情報処理装置110間で、前記サーバ情報処理装置120の前記磁気ディスク5bに格納されたファイルを共用する。ユーザーは前記共用を行うため、前記クライアントプログラム24を介して前記磁気ディスク5b上の共用ファイルをアクセスする。

【0026】また、クライアント情報処理装置110のICカードコントローラ6は、ICカード130が脱着可能となっている。前記ICカード130内には、CPU1c、不揮発性メモリ7、入出力インタフェース8が具備されている。前記不揮発性メモリ7には、アクセス制御プログラム30、グループ暗号プログラム28、ID情報29、マスタ鍵31が書き込まれている。前記不揮発性メモリ7の内容は、前記入出力インタフェース8を介して外部からアクセスすることができる。ただし、不正なアクセスから前記不揮発性メモリ7の記憶内容を保護する為、前記CPU1cによるユーザー認証制御が必ず介在する仕組みとなっている。

【0027】図27は、前記共用ファイル暗号プログラム21と前記ファイルI/Oフックプログラム22を構成するプログラムルーチンを示す図である。前記共用ファイル暗号プログラム21は、前記ICカード130へのログイン、ログアウトを行うログイン制御ルーチン1000、ログアウト制御ルーチン1100、ユーザーがファイルの暗号化をその都度明示的に指示し暗号化する手動ファイル暗号化ルーチン2000、前記手動ファイ

ル暗号化をHTML (Hyper Text Markup Language) で記述した共用ファイル向けにアレンジしたHTMLファイル暗号化ルーチン2100から構成される。前記ファイルI/Oフックプログラム22は、ファイルオープンフックルーチン1200、ファイルクリエイトフックルーチン1300、ファイルリードフックルーチン1400、ファイルライトフックルーチン1500、ファイルクローズフックルーチン1600、ファイルリネームフックルーチン1700から構成される。

【0028】図2は、前記ID情報29の内容の一例を示す図である。ここでは、カテゴリ毎の内容を示すデータとそのコードを10組まで登録できるようになっている。実際に前記不揮発性メモリ7に記憶するのは、データとコードの値だけである。

【0029】次に、上記グループ暗号プログラム28の動作について説明する。グループ暗号では、復号化を許す相手の宛先リストを作成し、該宛先リストをハッシュ関数に入力することによって得られるハッシュ値を、暗号化・復号化の鍵に用いる。前記鍵をグループ鍵51と呼ぶ。本実施例における宛先リストは、前記ID情報29を条件式で連結した形で表現する。具体的には、「カテゴリ番号」「データコード識別子」「演算子」「データ」を一組とし、カンマ(,)で区切って並べる。このカンマは論理和の意味を持つ。演算子としては、等しい(=)の他、大小(>、<)や論理積(^)を用いることができる。例えば、図2で示した前記ID情報29の定義を用いた場合、

「システム研究所の第四部の主任以上と暗号太郎」

という宛先に対する宛先リストは、

「5C=676^6C=04^8C>=5, 2D=暗号太郎」

となる。

【0030】図3は、前記宛先リストのハッシュ値を計算し前記グループ鍵51を作成する手順の一例を示す図である。31はマスタ鍵、41は宛先リスト、51はグループ鍵、32はハッシュ関数である。まず、前記ID情報を元に作成した宛先リストのデータ長が8の倍数のバイト数になっていない場合には不足分を空白文字で補い更に8バイトの乱数を付加する。宛先リストが丁度8の倍数であった場合には単に8バイトの乱数を付加する。8の倍数とした宛先リストを以下では単に宛先リスト41と呼ぶ。このように乱数を付加することによって、同じ宛先でも毎回異なるグループ鍵が生成されるのでセキュリティを向上することができる。

【0031】次にハッシュ関数32の動作を簡単に説明する。まず、前記宛先リスト41を先頭から8バイトずつブロック暗号方式によって暗号化し、該暗号化の結果と初期値との排他的論理和を取り次の段のブロック暗号の初期値として用いる。この処理を前記宛先リスト41

の最後まで繰り返し、最後の段から出力される暗号化結果と初期値との排他的論理和をグループ鍵51とする。なお、前記ブロック暗号では、8バイトのマスタ鍵31と8バイトの初期値を用いて入力データの暗号化を行う。なお、データを暗号化するユーザーと復号化するユーザーは、同じ値の前記マスタ鍵31を使用する。

【0032】図4は、前記ICカード130内の前記グループ暗号プログラム28の処理内容を示すフロー図である。グループ暗号プログラム28では、上述のグループ鍵生成方法を用いて前記宛先リスト41からグループ鍵51を作成する。ステップ401では、前記ICカード130のアクセス制御プログラム30に対するログインが完了しているかどうかを検査し、完了していなければステップ411でエラーコードを設定して本プログラムを終了する。前記ログインが完了している場合には、入力された宛先リスト41をステップ402で解析し、前記ID情報29に該当する宛先があるかどうかを検査する。該検査の結果宛先が該当していればステップ403を実行し、該当していなければステップ412でエラーコードを設定し本プログラムを終了する。ステップ403では、図3で説明した方法でグループ鍵を生成する。以上の処理によって、ICカード130にログイン済みで且つ前記宛先リストの宛先としてICカード130内のID情報29に該当するものがある場合のみグループ鍵を生成するように制御することができる。

【0033】以上説明したグループ鍵51を用いてデータを暗号化することによって、前記宛先リスト41に含まれるID情報29を持つ者だけが該データの復号化を可能とすることができる。

【0034】次に、上述のグループ暗号をファイルの暗号に適用した例について説明する。図5は、ファイルの暗号化手順と暗号化後のファイルの構成を示す図である。501は暗号化する前の平文ファイル、502は暗号化後の暗号文ファイルである。61は乱数として生成された実行鍵、504は前記実行鍵61を鍵として前記ファイル501の内容を暗号化した暗号文データ、505は暗号文データ504に付加するヘッダ、506から511は前記ヘッダ505の内容であり、506は使用した暗号化方式を示す文字列、507は前記平文ファイル501のファイル名、508前記平文ファイル501のファイルサイズ、509は前記実行鍵61をグループ鍵51で暗号化した暗号化実行鍵、510は前記宛先リスト41のサイズ、511は前記宛先リスト41を前記マスタ鍵31で暗号化した暗号化宛先リストである。前記暗号文ファイル502は、図6に示すように上記手順と全く逆の手順で復号化することができる。上述のファイル暗号化においては、前記グループ鍵51は前記実行鍵61を暗号化する為に用い、ファイルデータの暗号化には前記実行鍵61を用いた。したがって、前記宛先リスト41を変更する場合には、まず前記実行鍵61を変

更前の前記宛先リスト 4 1 で生成した前記グループ鍵 5 1 で復号化し、次に変更後の前記宛先リスト 4 1 から生成した前記グループ鍵 5 1 で再暗号化すればよい。これにより、前記宛先リスト 4 1 の変更時にも前記実行鍵 6 1 に比べデータ量の多いファイルデータは再暗号化が不要とすることができ、変更処理を高速に行うことができる。

【0035】次に、上記ファイルの暗号化を用いてユーザーが操作するファイルを自動的に暗号化・復号化する機能について説明する。ここで用いるファイルシステムは、最近の OS が具備する一般的なものであり、ツリー状の階層構造を持つ。ツリーの節の部分はディレクトリと呼ばれ、各ディレクトリに存在するファイルはディレクトリファイルと呼ばれるファイルによって管理される。ファイルの物理的な格納場所は、前記クライアント情報処理装置 1 1 0 の前記磁気ディスク 5 a あるいは前記サーバ情報処理装置 1 2 0 の前記磁気ディスク 5 b のどちらでもよい。

【0036】図 7 は、ユーザーから見た論理的なディレクトリの構成を示す図である。本実施例のファイルの自動暗号化、復号化を導入していない場合は、前記論理的なディレクトリ構成と物理的なディレクトリ構成は同一である。図 8 は、ファイルの自動暗号化、復号化を導入した後の物理的なディレクトリ構成を示す図である。ユーザーがファイルの自動暗号化、復号化を行うディレクトリには、ファイル名としてユーザー名とその末尾に「. dei」を付加したファイル名で宛先リストファイル 9 0 1 を作成する。また、全ユーザーに共通な宛先を設定して暗号化を行う場合には、「default. dei」というファイル名で宛先リストファイル 9 0 1 を作成する。このように、前記宛先リストファイル 9 0 1 を作成しておく、前記宛先リストファイル 9 0 1 に記載された宛先リストを宛先としたファイルの自動暗号化が行われ、連番とその末尾に「. s s f」を付加したファイル名で格納される。例えば、図 7 のファイル「osaka. doc」は「0001. s s f」として格納される。また前記自動暗号化が行われたディレクトリには、「fileinfo. dei」という名称の暗号ファイル情報ファイル 9 0 2 が作成される。

【0037】図 9 は、前記宛先リストファイル 9 0 1、前記暗号ファイル情報ファイル 9 0 2 の内容の一例を示す図である。各ユーザーの宛先リストファイル 9 0 1 は、該ユーザーを宛先としたグループ暗号で暗号化している。また、「default. dei」は、前記マスタ鍵 3 1 を用いて暗号化している。これによって、宛先リストファイルの不正な読み出しや改竄を防止できる。一方、暗号ファイル情報ファイル 9 0 2 には、暗号化後のファイル名、暗号化前の平文ファイル名、暗号化前の平文ファイルのサイズが書き込まれている。上述のファイル名称はあくまでも一例であり他の名称を用いてもよ

い。

【0038】以下、図 8 に示したような物理的構造を持つファイルを図 7 に示した論理的ファイル構造として扱うための処理プログラムの詳細を説明する。

【0039】まず、共用ファイル暗号プログラム 2 1 内のログイン処理ルーチン 1 0 0 0、ログアウト処理ルーチン 1 1 0 0 について説明する。図 1 0 は、前記 IC カード 1 3 0 へのログイン処理ルーチン 1 0 0 0 の処理フローを示す図である。まず、ステップ 1 0 0 1 でユーザーが ID とパスワードの入力を行い、ステップ 1 0 0 2 では前記ユーザー ID と前記パスワードを用いて前記 IC カード 1 3 0 にログインを試み、ステップ 1 0 0 3 で前記ログインが成功したかどうかを判定する。前記判定の結果、失敗の場合はステップ 1 0 0 9 を実行し、3 回続けて失敗するまではステップ 1 0 0 1 以下を繰り返すよう制御する。ログインに成功すると、ステップ 1 0 0 4 でイベント待ち状態となる。イベントが到着すると、ログアウト要求かどうかを調べ、ログアウト要求ならばステップ 1 0 0 8 でログアウト処理を行い終了する。前記イベントがログアウト要求でなければステップ 1 0 0 6 でタイマイイベントかどうかを調べ、タイマイイベントでなければステップ 1 0 0 4 に戻る。タイマイイベントであった場合には、IC カード 1 3 0 が前記クライアント情報処理装置 1 1 0 に接続されているかどうかを調べ、接続されていればステップ 1 0 0 4 に戻り、接続されていない場合は、ステップ 1 0 0 8 のログアウト処理を実行し、終了する。このように、タイマイイベントを用いて定期的に前記 IC カード 1 3 0 の接続を調べることによって前記 IC カード 1 3 0 を抜いた際のログアウト処理を自動化することができる。

【0040】図 1 1 は、前記 IC カード 1 3 0 からのログアウト処理ルーチン 1 1 0 0 の処理フローを示す図である。まず、ステップ 1 1 0 1 でオープン中の暗号ファイルがあるかどうかを調べ、あればステップ 1 1 0 2 でユーザーにログアウトを中止するか否かを尋ねる。ログアウト処理を中止する場合には、本ルーチンを終了する。中止しない場合には、ステップ 1 1 0 3 でオープン中の暗号ファイルのクローズ処理を行い、処理を継続する。ステップ 1 1 0 4 では、IC カード 1 3 0 がクライアント情報処理装置 1 1 0 に接続されているかどうかを調べ、接続されていればステップ 1 1 0 5 で IC カード 1 3 0 に対してログアウトコマンドを送出し処理を終了する。接続されていなければそのまま処理を終了する。

【0041】次に、前記ファイル I/O フックプログラム 2 2 内の各ルーチンの処理について説明する。まず、ファイルのデータ入出力とその管理方法から説明する。

【0042】図 1 8 は、本実施例のファイル暗号化・復号化におけるデータの入出力方法を示す図である。前記暗号文ファイル 5 0 2 の前記暗号文データ 5 0 4 は、暗号文ブロックバッファ 8 2 のサイズ単位で前記暗号文ブ

ロックバッファ82との間で読み出し・書き込みが行われる。前記暗号文ブロックバッファ82のデータは復号化し平文ブロックバッファ81に転送される。また、前記平文ブロックバッファ81のデータは暗号化し前記暗号文ブロックバッファ82に転送される。

【0043】図19は、オープン中の暗号ファイルを管理するためのファイルハンドルテーブル90である。ハンドル番号1902、オープンモード1903、平文ファイルポインタ1904、暗号ファイルポインタ1905、バッファポインタ1906、更新フラグ1907、宛先リスト41の各情報を、オープン中の各ファイル毎にポインタ1901を用いてリスト構造で記憶している。ここでハンドル番号1902は、ファイルをオープンした際に前記OS20から与えられる管理番号、オープンモード1903とは、読み出し専用、書き込み専用、あるいは読み書き用などのモードを示す値である。前記平文ファイルポインタ1904は復号化後の平文ファイル501に対するリード・ライトを行う際の起点を示すものであり、平文ファイル501の先頭からのバイト数で示される。なお、平文ファイル501は、平文ブロックバッファ81を通して読み書きする論理的なファイルとして存在する。暗号ファイルポインタ1905は、暗号文ファイルに対するリード・ライトを行う際の起点を示すものであり、502のヘッダ505を含む先頭からのバイト数で示される。前記暗号ファイルポインタ1905は、ファイルの最後のブロック部分以外ではバッファサイズ単位で増減する。バッファポインタ1906は、平文ブロックバッファ81および暗号文ブロックバッファ82に転送されているデータの先頭が前記平文ファイル501のどの部分に該当するデータであるかを示す値であり、前記平文ファイル501の先頭からのバイト数が格納してある。前記バッファポインタ1906はバッファサイズの単位でのみ増減する。更新フラグ1907は、前記平文ブロックバッファ81のデータが、ライトI/O操作によって更新されているが暗号文ブロックバッファ82への転送と暗号文ファイル502への書き込みが完了していない状態を示すフラグである。

【0044】図12は、前記ファイルオープンルーチン1200の処理フローを示す図である。まず、ステップ1281で、フックしたファイルのディレクトリに前記暗号ファイル情報ファイル902が存在するか否かを調べ、存在すれば前記暗号ファイル情報ファイル902の内容を参照し、フックしたファイルのファイル名が登録されているか否かを調べる。登録されていれば、ステップ1282で対応する暗号文ファイル502のファイル名称にファイル名称を変更する。次に、ステップ1201で前記OS20aに対しファイルオープン要求を発行する。該オープンが成功したかどうかをステップ1202で調べ、失敗した場合は本ルーチンを終了す

る。次にステップ1203で、ICカード130へのログインが終了しているかどうかを調べ、ログイン済みの場合は、ステップ1207に処理を移す。ログイン済みでなかった場合には、ステップ1204で当該ファイルが暗号ファイルであるかどうかを調べ、暗号ファイルでなかった場合は本ルーチンの処理を終了する。暗号ファイルの場合には、ステップ1205で前記ログイン処理ルーチン1000を呼び出すことによってICカード130へのログイン処理を行う。ステップ1206では、前記ログイン処理が成功したかどうかを調べ、成功しなかった場合はステップ1215でメッセージ21252を表示して本ルーチンを終了する。ログインに成功した場合は、ステップ1207で当該ファイルの復号権限があるか、即ち宛先にログイン中のユーザーが含まれているかどうかを判定する。前記判定は暗号ファイルの前記ヘッダ505の前記暗号化宛先リスト511を復号化することによって前記宛先リスト41を取得し、該宛先リスト41を前記ICカード130の前記グループ暗号プログラム28に入力し、グループ鍵51が出力されるかエラーコードが出力されるかで行う。復号権限がない場合には、ステップ1215でメッセージ21252を表示して本ルーチンを終了する。復号権限がある場合にはステップ1209において、当該ユーザーまたは共通の前記宛先リストファイル901が当該ディレクトリに存在するかどうかを調べ、存在すれば暗号化対象ディレクトリであると判定する。暗号化対象ディレクトリでなければステップ1216を実行し、フックしているファイルが暗号ファイルであるかどうかを調べ、暗号ファイルでなければ本ルーチンを終了する。暗号ファイルの場合には、ステップ1217でメッセージ31253を表示し、処理を中断するか否かをユーザーに問い合わせる。上記ステップ1209で暗号化対象ディレクトリと判定した場合は、ステップ1210でフックしているファイルが暗号ファイルかどうかを調べ、暗号ファイルでなかった場合には、ステップ1211でメッセージ11251を表示し、処理を中断するか否かをユーザーに問い合わせる。ステップ1210の判定で、暗号ファイルであった場合には、ステップ1218で当該ファイルの宛先リスト41と当該ディレクトリの宛先リストファイル901に書かれた宛先リスト41の宛先が同じか異なるかを調べ、異なっていた場合には、ステップ1219でメッセージ41254を表示し、処理を中断するか否かをユーザーに問い合わせる。前記ステップ1218で宛先が同じと判定した場合には、ステップ1220でファイルハンドルテーブル90への登録を行い、本ルーチンを終了する。また、前記ステップ1217、1211、1219でのメッセージ表示に対し、ユーザーが処理の中止を選択したかどうかはステップ1212で判定し、中止の場合はステップ1213でファイルをクローズし、ステップ1214でリターンコードにオープン

エラーを設定し本ルーチンを終了する。前記ステップ 1 2 1 2 での前記判定が中止でなければ、前記ステップ 1 2 2 0 を実行し、本ルーチンを終了する。

【0045】図 1 3 は、ファイルクリエイトフックルーチン 1 3 0 0 の処理フローを示す図である。まず、ステップ 1 3 0 1 で、IC カード 1 3 0 へのログインが終了しているかどうかを調べ、ログイン済みの場合はステップ 1 3 0 6 に処理を移す。ログイン済みでなかった場合には、ステップ 1 3 0 2 で当該ユーザーまたは共通の前記宛先リストファイル 9 0 1 が当該ディレクトリに存在するかどうかを調べ、存在すれば暗号化対象ディレクトリであると判定する。暗号化対象ディレクトリでなければ、ステップ 1 3 3 1 でファイルクリエイトを実行し本ルーチンを終了する。暗号化対象ディレクトリの場合、ステップ 1 3 0 3 でメッセージ 5 1 3 5 5 を表示し、ステップ 1 3 0 4 で前記ログイン処理ルーチン 1 0 0 0 を呼び出すことによって前記 IC カード 1 3 0 へのログイン処理を行う。ステップ 1 3 0 5 では、前記ログイン処理が成功したかどうかを調べ、成功しなかった場合はステップ 1 3 4 1 でメッセージ 6 1 3 5 6 を表示してファイルクリエイトを中止するか否かユーザーに問い合わせる。ステップ 1 3 4 2 でユーザーの選択を判定し、中止の場合はステップ 1 3 4 3 でクリエイトエラーを設定し本ルーチンを終了する。中止でない場合は、ステップ 1 3 2 0 でファイルクリエイトを実行し本ルーチンを終了する。一方、ステップ 1 3 0 5 でログインに成功した場合は、ステップ 1 3 0 6 で、当該ディレクトリに当該ユーザーまたは共通の前記宛先リストファイル 9 0 1 が存在するかどうかを調べ、存在すれば暗号化対象ディレクトリであると判定する。暗号化対象ディレクトリでなければステップ 1 3 2 0 でファイルクリエイトを実行し、本ルーチンを終了する。暗号化対象ディレクトリである場合は、ステップ 1 3 0 7 でファイル名称を、連番とその末尾に「. s s f」を付加したファイル名に置き換え、ステップ 1 3 0 8 でファイルクリエイトを実行する。ステップ 1 3 0 9 では、前記ファイルクリエイトが成功したかどうかを調べ、失敗ならば本ルーチンを終了し、成功ならばステップ 1 3 1 0 以下を実行する。ステップ 1 3 1 0 では、暗号ファイルの前記ヘッダ 5 0 5 を生成し、ステップ 1 3 0 8 で作成したファイルの先頭部分に書き込む。続くステップ 1 3 1 1 では前記暗号ファイル情報ファイル 9 0 2 への登録を行い、さらにステップ 1 3 1 2 でファイルハンドルテーブル 9 0 への登録を行い本ルーチンを終了する。

【0046】図 1 4 は、リードフックルーチン 1 4 0 0 の処理フローを示す図である。まず、ステップ 1 4 0 1 でリード I / O のパラメータに含まれるファイルハンドル番号が前記ファイルハンドルテーブル 9 0 に既に登録されているかどうかを調べ、登録されていない場合は暗号・復号対象のファイルではないのでステップ 1 4 2 1 で

通常のファイルリード I / O を実行し、本ルーチンを終了する。ファイルハンドルテーブル 9 0 に登録済みの場合は、ステップ 1 4 0 2 で、前記ファイルハンドルテーブル 9 0 の前記オープンモード 1 9 0 3 がリード可になっているかどうかを調べ、リード不可であれば処理が継続できないので、ステップ 1 4 2 2 でエラーコードを設定して本ルーチンを終了する。次に、ステップ 1 4 0 3 で前記平文ファイルポインタ 1 9 0 4 を取得する。ステップ 1 4 0 4 では、前記平文ファイルポインタ 1 9 0 4 が、前記バッファポインタ 1 9 0 6 から始まるバッファの中を指し示しているかどうかを調べる。指し示していればステップ 1 4 1 3 に処理を移し、指し示していなければステップ 1 4 0 5 以下を実行する。ステップ 1 4 0 5 では前記更新フラグ 1 9 0 7 を参照し、前記平文ブロックバッファ 8 1 中に更新データがあるかどうかを調べ、あればステップ 1 4 0 6 で実際の暗号文ファイル 5 0 2 のファイルポインタを OS へのシークコマンドによって前記バッファポインタ 1 9 0 6 へと移動させる。ステップ 1 4 0 7 で平文ブロックバッファ 8 1 のデータを暗号化し、暗号文ブロックバッファ 8 2 に転送すると共に、暗号文ファイル 5 0 2 に書き戻す。ステップ 1 4 0 8 では前記更新フラグ 1 9 0 7 をクリアし、前記平文ブロックバッファ 8 1 中に更新データがないことを示すように設定する。ステップ 1 4 0 9 で実際の暗号文ファイル 5 0 2 のファイルポインタを OS へのシークコマンドによって前記暗号ファイルポインタ 1 9 0 5 へと移動させる。ステップ 1 4 1 0 では、ブロックサイズ分のデータを前記暗号文ファイル 5 0 2 から暗号文ブロックバッファ 8 2 へと読み出す。ステップ 1 4 1 1 では、前記暗号文ファイルポインタ 1 9 0 5、前記バッファポインタ 1 9 0 6 を更新する。ステップ 1 4 1 2 では、暗号文ブロックバッファ 8 2 のデータを復号化し平文ブロックバッファ 8 1 に転送する。ステップ 1 4 1 3 では、フックしたファイルリード I / O の要求元アプリケーションプログラム 2 3 が指定した要求バッファに平文ブロックバッファ 8 1 からデータを転送する。ステップ 1 4 1 4 では、前記転送を行ったバイト数を前記平分ファイルポインタ 1 9 0 4 に加算する。ステップ 1 4 1 5 では、フックしたファイルリード I / O の要求元が要求したサイズ分のデータを前記要求バッファに転送したかどうかを調べ、未完了ならばステップ 1 4 1 0 以下を再び実行し、完了ならば本ルーチンを終了する。

【0047】図 1 5 は、ライトフックルーチン 1 5 0 0 の処理フローを示す図である。まず、ステップ 1 5 0 1 でフックしたライト I / O のパラメータに含まれるファイルハンドル番号が前記ファイルハンドルテーブル 9 0 に既に登録されているかどうかを調べ、登録されていない場合は暗号・復号対象のファイルではないのでステップ 1 5 2 1 で通常のファイルリード I / O を実行し、本ルーチンを終了する。ファイルハンドルテーブル 9 0 に登

録済みの場合は、ステップ1502で、前記ファイルハンドルテーブル90の前記オープンモード1903が「リードライト可」になっているかどうかを調べ、「リードライト可」でなければステップ1531でエラーコードを設定して本ルーチンを終了する。「ライト可」だけではなく「リードライト可」の必要があるのは、暗号化のブロック単位よりも小さなサイズの書込に対しては、不足分のデータをリードした上で暗号化して書き戻すリードモディファイライト処理が必要なためである。前記ファイルオープンフックルーチンで「ライト可」を強制的に「リードライト可」に置き換えてファイルをオープンしておいてもよい。次に、ステップ1503で前記平文ファイルポインタ1904を取得する。ステップ1504では、前記平文ファイルポインタ1904が、前記バッファポインタ1906から始まるバッファの中を指し示しているかどうかを調べる。指し示していればステップ1513に処理を移し、指し示していなければステップ1505以下を実行する。ステップ1505では前記更新フラグ1907を参照し、前記平文ブロックバッファ81中に更新データがあるかどうかを調べ、更新データがなければステップ1510に処理を移し、更新データがあればステップ1506で実際の暗号文ファイル502のファイルポインタをOSへのシークコマンドによって前記バッファポインタ1906へと移動させる。ステップ1507では、平文ブロックバッファ81のデータを暗号化し、暗号文ブロックバッファ82に転送すると共に、暗号文ファイル502に書き戻す。ステップ1508では前記更新フラグ1907をクリアし、前記平文ブロックバッファ81中に更新データがないことを示すように設定する。ステップ1509では、暗号文ファイル502のファイルポインタをOSへのシークコマンドによって前記暗号ファイルポインタ1905が指し示す位置へと移動させる。ステップ1510では、フックしたファイルライトI/O要求の書込データ領域が、現在の平文ブロックバッファ81全体を包含するかどうかを調べる。前記調査の結果全体を包含していない場合は、不足分を読み出しマージする必要があるので、ステップ1511で暗号文ブロックバッファ82のサイズ分だけ暗号文ファイル502をリードする。リードしたデータは、前記暗号文ブロックバッファ82に格納すると共に、復号化し前記平文ブロックバッファ81に格納する。一方、前記ステップ1510での調査の結果、全体を包含していれば不足分を読み出す必要がないのでステップ1518でステップ1511のリードサイズと同じサイズ分だけシーク処理を行う。ステップ1512では、暗号文ファイルポインタ1905、バッファポインタ1906を更新する。ステップ1513では、平文ファイルポインタ1904の指し示す位置から暗号文ファイルポインタ1905の指し示す位置の手前までのデータを、フックしたファイルI/O要求の書込データバッ

ファから平文ブロックバッファ81に転送する。ステップ1514では、平文ブロックバッファ81の内容が更新されたことを示す更新フラグ1907をセットし、ステップ1515でフックしたファイルI/Oライト要求の全データの転送を完了したかどうかを判定し、完了していなければステップ1506以下を完了するまで繰り返し実行する。以上の処理によって、平文ブロックバッファ81は、先読みと先復号化キャッシュとして動作し、且つライトバックキャッシュとして動作するので、暗号化・復号化によるファイルI/O処理速度の低下を少なく抑えることができる。

【0048】図16は、クローズフックルーチン1600の処理フローを示す図である。まず、ステップ1601で前記ファイルハンドルテーブル90に、当該クローズ要求のあったファイルハンドルが登録されているかどうかを調べ、登録されていない場合はステップ1607に処理を移行し、ファイルをクローズして終了する。前記ファイルハンドルテーブル90に登録済みのファイルであった場合には、前記ファイルハンドルテーブル90の更新フラグ1907を参照し、前記更新フラグ1907がセット、即ちファイルに書き戻されていない更新データが平文ブロックバッファ81に存在するかどうかを調べる。前記更新フラグ1907がセットされていた場合には、ステップ1603で平文ブロックバッファ81の内容を暗号化し、ファイルに書き戻す処理を行う。続くステップ1604では、前記平文ブロックバッファ81を、全て0などのデータを書き込むことによってクリアし、ステップ1605で前記平文ブロックバッファ81と前記暗号文ブロックバッファ82を解放する。ステップ1606では、前記ファイルハンドルテーブル90から当該ファイルのノードを削除し、ステップ1607でファイルクローズ要求をOSに発行し、本ルーチンを終了する。

【0049】図17は、ファイルリネームフックルーチン1700の処理フローを示す図である。まず、ステップ1701でリネームの対象がディレクトリファイルかどうかを判定し、ディレクトリファイルならばステップ1721でリネームを実行し、本ルーチンを終了する。ディレクトリファイルでなければ、ステップ1702で対象ファイルが暗号ファイルかどうかを判定し、暗号ファイルでなければ前記ステップ1721でリネームを実行し、本ルーチンを終了する。続くステップ1703では、当該ファイルの復号権限があるか、即ち前記ヘッダ505の宛先リスト41にログイン中のユーザーのID情報29とマッチする項目が含まれているかどうかを判定し、復号権限がなければ、ステップ1711で権限がない旨のメッセージを表示し、ステップ1712でエラーコードを設定し本ルーチンを終了する。復号権限がある場合は、ステップ1704で、当該ファイルのヘッダ505の平文ファイル名507を前記アプリケーション



プログラム 2 3 が指示した新名称に変更し、ステップ 1 7 0 5 では暗号ファイル情報ファイル 9 0 2 に記載された当該ファイルの平文ファイル名称を前記ステップ 1 7 0 4 と同様に新名称に変更し、本ルーチンを終了する。

【0 0 5 0】以上説明したファイル I / O フックプログラム 2 2 の各ルーチンの処理によって、所望のディレクトリ内のファイルの暗号化・復号化をアプリケーションプログラム 2 3 からは透過に実現することができる。

【0 0 5 1】次に、前記共用ファイル暗号プログラム 2 1 に含まれる手動ファイル暗号化ルーチン 2 0 0 0 について図 2 0 を用いて説明する。このルーチンは、暗号化を行う平文ファイルのファイル名称と、暗号化後の暗号文ファイル名称と宛先リスト 4 1 を入力として動作する。まず、ステップ 2 0 0 1 では、指定された平文ファイルをオープンし、ステップ 2 0 0 2 で仮の名称を付けた一時ファイルを作成する。ステップ 2 0 0 4 では指定された前記宛先リスト 4 1 を前記 I C カード 1 3 0 のグループ暗号プログラム 2 8 に与えることによってグループ鍵を生成する。前記生成には前記 I C カード 1 3 0 へのログインが必要であるが、上記の実施例と同様であるのでここでは記述を省略している。ステップ 2 0 0 5 では乱数である前記実行鍵 4 1 を生成し、ステップ 2 0 0 6 で前記ヘッダ 5 0 5 を前記一時ファイルの先頭部分に書き込む。ステップ 2 0 0 9 では、平文ファイルからデータを読み出し、ステップ 2 0 1 0 で該データを暗号化し、ステップ 2 0 1 1 で該暗号化データを一時ファイルに書き込む。ステップ 2 0 1 2 では、前記平文ファイルの全データの暗号化を終了したかどうかを判定し、終了するまで前記ステップ 2 0 0 9、2 0 1 0、2 0 1 1 を繰り返し実行するように制御する。ステップ 2 0 1 3 では、前記平文ファイルと前記一時ファイルの両方をクローズする。ステップ 2 0 1 4 では、指定された平文ファイル名と暗号文ファイル名が同一かどうかを判定し、同一ならばステップ 2 0 1 5 で前記平文ファイルを削除する。ステップ 2 0 1 6 では、前記一時ファイルのファイル名称を指定された暗号文ファイル名称に変更し、本プログラムを終了する。

【0 0 5 2】次に、HTML ファイルの暗号化を行う実施例について説明する。図 2 2 の 2 2 0 1 は、平文の HTML ファイルの一例である。前記 HTML ファイルは、HTML ブラウザで表示すると図 2 3 に示す画面となる。図 2 4 の 2 4 0 1 は、暗号化した HTML ファイルの一例であり、リーダ部分 2 4 0 2、前記平文の HTML ファイル 2 2 0 1 を暗号化した本体部分 2 4 0 3、フッタ部分 2 4 0 4 を合成したものである。リーダ部分 2 4 0 2 の最後は HTML でのコメント文開始を示す「< ! —」が、フッタ部分 2 4 0 4 の最初は HTML でのコメント文終了を示す「— >」が設定されており、前記合成によって本体部分 2 4 0 3 はコメント文の内容となる。HTML ブラウザによる表示において、前

記コメント文の内容は表示されないで、前記の暗号化 HTML ファイル 2 4 0 1 を HTML ブラウザで表示した結果は、図 2 5 に示すような画面となる。このように、暗号化した結果をコメント文として埋め込むことによって、復号化プログラムを具備しないクライアント情報処理装置 1 1 0 で暗号化ファイルを誤って表示した場合にも、意味のない文字列を画面に表示することを防止することができる。

【0 0 5 3】図 2 1 は、上記の HTML ファイル暗号化を行う HTML ファイル暗号化プログラムの処理フローを示す図である。ステップ 2 1 0 1 では、指定された平文の HTML ファイル 2 2 0 1 をオープンし、ステップ 2 1 0 2 で仮の名称を付けた一時ファイルを作成する。ステップ 2 1 0 4 では指定された前記宛先リスト 4 1 を前記 I C カード 1 3 0 のグループ暗号プログラム 2 8 に与えることによってグループ鍵を生成する。ステップ 2 1 0 5 ではリトライカウンタを初期化し、ステップ 2 1 0 6 では乱数である前記実行鍵 4 1 を生成する。ステップ 2 1 0 7 では前記ヘッダ 5 0 5 を前記一時ファイルの先頭部分に書き込む。ステップ 2 1 0 9 では、前記リーダ部分 2 4 0 2 を前記一時ファイルの前記ヘッダ 5 0 5 の続きに書き込む。ステップ 2 1 1 0 では、前記平文の HTML ファイル 2 2 0 1 からデータを読み出し、ステップ 2 1 1 1 で該データを暗号化する。ステップ 2 1 1 2 では、暗号化後のデータの中に前記コメントの終了文字列「— >」がないかどうかを調べ、あればステップ 2 1 3 1 以下を実行し、なければステップ 2 1 1 3 で前記一時ファイルに前記暗号化後のデータを書き込む。ステップ 2 1 1 4 では、前記平文の HTML ファイル 2 2 0 1 の全データの暗号化を終了したかどうかを判定し、終了するまで前記ステップ 2 1 1 0、2 1 1 1、2 1 1 2、2 1 1 3 を繰り返し実行するように制御する。ステップ 2 1 1 5 では、前記フッタ部分 2 4 0 4 を前記一時ファイルに書き込み、ステップ 2 1 1 6 で前記平文の HTML ファイル 2 2 0 1 と前記一時ファイルの両方をクローズする。ステップ 2 1 1 7 では、指定された平文の HTML ファイル名と暗号化後の HTML ファイル名が同一かどうかを判定し、同一ならばステップ 2 1 1 8 で前記平文の HTML ファイル 2 2 0 1 を削除する。ステップ 2 1 1 9 では、前記一時ファイルのファイル名称を指定された暗号化後の HTML ファイル名称に変更し、本プログラムを終了する。一方、前記ステップ 2 1 1 2 において、暗号化後のデータの中に前記コメントの終了文字列「— >」があると判定された場合は、ステップ 2 1 3 1 で前記一時ファイルに書き込んだ内容をクリアする。次に、ステップ 2 1 3 2 で前記リトライカウンタをカウントアップし、ステップ 2 1 3 3 で前記リトライカウンタが予め設定した上限値（本実施例では 1 0 回とした）を越えていないかを検査し、越えていなければステップ 2 1 0 6 からの部分を再度実行する。この再実行で



は、前記ステップ 2 1 0 6 で前回とは異なる乱数が実行鍵 6 1 として使われるので、前記ステップ 2 1 1 1 の暗号化で再び前記コメントの終了文字列「—>」が現れる確率はかなり低い。それにも拘わらず、前記上限値を越えてしまった場合は、ステップ 2 1 3 4 で前記平文の HTML ファイル 2 2 0 1 と前記一時ファイルの両方をクローズする。さらにステップ 2 1 1 5 で前記一時ファイルを削除し、ステップ 2 1 3 6 でエラーメッセージを表示し、本プログラムを終了する。なお、暗号化したデータはランダムな値となるので、前記コメントの終了文字列「—>」が現れる確率は、一文字は 1 バイト (= 8 ビット) で表現されるので「(2 の 8 乗) の 3 乗」分の 1、即ち 1 6 7 7 7 2 1 6 分の 1 である。言い換えれば約 1 6 M バイトのデータに一回起こる程度ということである。したがって、上述のように実行鍵 6 1 を替えてリトライする制御を組み込んでおけば、実用上全く問題がないことが判る。また、前記コメントの終了文字列「—>」が前記本体部分 2 4 0 3 に出現することを避ける別の手段として、暗号化したデータを 1 6 進表示によるテキスト文字列として格納することもできる。

【0 0 5 4】この方法で作成した暗号化 HTML ファイルを図 2 6 に示す。ただし、この方法では、1 バイトのデータを表現するために 2 文字即ち 2 バイトが必要であり、データ量が倍になってしまうという問題もある。

【0 0 5 5】本実施例では、HTML で記述されたドキュメントファイルについて述べたが、解釈時に無視されるコメント文を記述するための書式を備えた文法であれば同様に本発明を適用することができることは明らかである。

【0 0 5 6】

【実施例】以下、本発明をインターネットに用いた場合の実施例を図を用いて説明する。まず、全体のシステム構成を説明する。図 1 は、本発明の暗号ファイル受信システム及びその制御方法の一構成例である。1 1 0 はクライアント情報処理装置、1 2 0 はサーバ情報処理装置である。

【0 0 5 7】サーバ情報処理装置 1 2 0 には、中央処理装置 (CPU) 1 b、メモリ 2 b、LAN コントローラ 3 b、ディスクコントローラ 4 b、磁気ディスク 5 b が具備されている。サーバ情報処理装置 1 2 0 の起動時には、オペレーティングシステム (OS) 2 0 b と、WWW (World-Wide Web) サーバプログラム 2 6、および共用ファイル暗号プログラム 2 1 b が、ディスクコントローラ 4 b を介して磁気ディスク 5 b からメモリ 2 b 上にロードされる。クライアント情報処理装置 1 1 0 には、CPU 1 a、メモリ 2 a、LAN コントローラ 3 a、ディスクコントローラ 4 a、磁気ディスク 5 a、IC カードコントローラ 6 がそれぞれ具備されている。クライアント情報処理装置 1 1 0 の起動時には、OS 2 0 a、共用ファイル暗号プログラム 2 1 a、通信

API (Application Program Interface) フックプログラム 2 7 が、前記ディスクコントローラ 4 a を介して前記磁気ディスク 5 a から前記メモリ 2 a 上にロードされる。また、ブラウザプログラム 1 9 は、前記クライアント情報処理装置 1 1 0 のユーザー (以下単にユーザーと呼ぶ) の指示によって前記ディスクコントローラ 4 a を介して前記磁気ディスク 5 a から前記メモリ 2 a 上にロードされる。前記通信 API フックプログラム 2 7 は、ブラウザプログラム 1 9 が前記 OS 2 0 a に発行する通信 API をフック (横取り) し、受信ファイルデータの復号化をバックグラウンドで自動的に行うものである。

【0 0 5 8】前記サーバ情報処理装置 1 2 0 と前記クライアント情報処理装置 1 1 0 は、ローカルエリアネットワーク (LAN) 1 0 1 を介して相互に接続されており、各々の LAN コントローラ 3 を介して通信を行う。なお、前記サーバ情報処理装置 1 2 0 と前記クライアント情報処理装置 1 1 0 は互いに遠隔地に存在し、電話回線や専用回線を介して接続されていてもよい。

【0 0 5 9】図 2 8 では一台のクライアント情報処理装置 1 1 0 を示したが、実際には、同じ構成のクライアント情報処理装置 1 1 0 が複数前記 LAN 1 0 1 に接続されており、これらのクライアント情報処理装置 1 1 0 間で、前記サーバ情報処理装置 1 2 0 の前記磁気ディスク 5 b に格納された HTML ファイルやイメージデータファイルを共用する。ユーザーは前記共用を行うため、前記ブラウザプログラム 1 9 を使用して前記磁気ディスク 5 b 上のファイルをアクセスする。なお、前記 HTML ファイルやイメージデータファイルは、前記サーバ情報処理装置 1 2 0 上で作成して前記磁気ディスク 5 b に格納しても良いし、クライアント情報処理装置 1 1 0 上で作成してから磁気ディスク 5 b に格納しても良い。また、クライアント情報処理装置 1 1 0 の IC カードコントローラ 6 は、IC カード 1 3 0 が脱着可能となっている。前記 IC カード 1 3 0 内には、CPU 1 c、不揮発性メモリ 7、入出力インタフェース 8 が具備されている。前記不揮発性メモリ 7 には、アクセス制御プログラム 3 0、グループ暗号プログラム 2 8、ID 情報 2 9、マスタ鍵 3 1 が書き込まれている。前記不揮発性メモリ 7 の内容は、前記入出力インタフェース 8 を介して外部からアクセスすることができる。ただし、不正なアクセスから前記不揮発性メモリ 7 の記憶内容を保護する為、前記 CPU 1 c によるユーザー認証制御が必ず介在する仕組みとなっている。

【0 0 6 0】図 2 9 は、前記共用ファイル暗号プログラム 2 1 と前記通信 API フックプログラム 2 7 を構成するプログラムルーチンを示す図である。前記共用ファイル暗号プログラム 2 1 は、前記 IC カード 1 3 0 へのログイン、ログアウトを行うログイン制御ルーチン 1 0 0 0、ログアウト制御ルーチン 1 1 0 0、ユーザーがファ

イルの暗号化をその都度明示的に指示し暗号化する手動ファイル暗号化ルーチン 2 0 0 0、前記手動ファイル暗号化を HTML (Hyper Text Markup Language) で記述した共用ファイル向けにアレンジした HTML ファイル暗号化ルーチン 2 1 0 0 から構成される。前記通信 API フックプログラム 2 7 は、send フックルーチン 1 8 0 0、recv フックルーチン 1 8 1 0、close socket フックルーチン 1 8 2 0 から構成される。

【0 0 6 1】前記 ID 情報 2 9 の内容は前記第 1 の実施例における図 2 に示した ID 情報 2 9 と同じである。ここでは、カテゴリ毎の内容を示すデータとそのコードを 1 0 組まで登録できるようになっている。実際に前記不揮発性メモリ 7 に記憶するのは、データとコードの値だけである。

【0 0 6 2】次に、上記グループ暗号プログラム 2 8 の動作は前記実施例におけるグループ暗号プログラム 2 8 の動作と、宛先リストのハッシュ値を計算し前記グループ鍵 5 1 を作成する手順は前記実施例における図 3 に示した手順とそれぞれ同じである。前記 IC カード 1 3 0 内の前記グループ暗号プログラム 2 8 の処理内容は前記実施例における図 4 に示したフロー図と同じである。以上説明したグループ鍵 5 1 を用いてデータを暗号化することによって、前記宛先リスト 4 1 に含まれる ID 情報 2 9 を持つ者だけが該データの復号化を可能とすることができる。

【0 0 6 3】次に、上述のグループ暗号をファイルの暗号に適用した例について説明する。図 3 0 は、ファイルの暗号化手順と暗号化後のファイルの構成を示す図である。図 5 に示した例とは、暗号化後のファイル 5 0 2 のヘッダ 5 0 5 の内容が相違している。すなわち、5 0 1 は暗号化する前の平文ファイル、5 0 2 は暗号化後の暗号ファイルである。6 1 は乱数として生成された実効鍵、5 0 4 は前記実効鍵 6 1 を鍵として前記ファイル 5 0 1 の内容を暗号化した暗号文データ、5 0 5 は暗号文データ 5 0 4 に付加するヘッダ、5 0 6 から 5 1 1 は前記ヘッダ 5 0 5 の内容であり、5 0 6 は使用した暗号化方式を示す文字列、5 0 9 は前記実効鍵 6 1 をグループ鍵 5 1 で暗号化した暗号化実効鍵、5 1 1 は前記宛先リスト 4 1 を前記マスタ鍵 3 1 で暗号化した暗号化宛先リスト、5 1 0 は前記暗号化宛先リスト 5 1 1 のサイズである。前記暗号ファイル 5 0 2 は、図 3 1 に示すように上記手順と全く逆の手順で復号化することができる。

【0 0 6 4】上述のファイル暗号化においては、前記グループ鍵 5 1 は前記実効鍵 6 1 を暗号化する為に用い、ファイルデータの暗号化には前記実効鍵 6 1 を用いた。したがって、前記宛先リスト 4 1 を変更する場合には、まず前記実効鍵 6 1 を変更前の前記宛先リスト 4 1 で生成した前記グループ鍵 5 1 で復号化し、次に変更後の前記宛先リスト 4 1 から生成した前記グループ鍵 5 1 で再

暗号化すればよい。これにより、前記宛先リスト 4 1 の変更時にも前記実効鍵 6 1 に比べデータ量の多いファイルデータは再暗号化が不要とすることができ、変更処理を高速に行うことができる。

【0 0 6 5】次に、共用ファイル暗号プログラム 2 1 内のログイン処理ルーチン 1 0 0 0、ログアウト処理ルーチン 1 1 0 0 について説明する。前記 IC カード 1 3 0 へのログイン処理ルーチン 1 0 0 0 の処理フローは、図 1 0 に示した第 1 の実施例におけるログイン処理ルーチン 1 0 0 0 と同じであり、詳細な説明は省略する。タイマイイベントを用いて定期的に前記 IC カード 1 3 0 の接続を調べることによって前記 IC カード 1 3 0 を抜いた際のログアウト処理を自動化することができる。

【0 0 6 6】前記 IC カード 1 3 0 からのログアウト処理ルーチン 1 1 0 0 の処理フローは、図 1 1 に示した第 1 の実施例におけるログアウト処理ルーチン 1 1 0 0 と同じであり、詳細な説明は省略する。

【0 0 6 7】図 3 2 は、本実施例のファイル暗号化・復号化におけるデータの入出力方法を示す図である。前記暗号ファイル 5 0 2 の前記暗号文データ 5 0 4 は、暗号文ブロックバッファ 8 2 のサイズ単位で前記暗号文ブロックバッファ 8 2 との間で読み出し・書き込みが行われる。前記暗号文ブロックバッファ 8 2 のデータは復号化し平文ブロックバッファ 8 1 に転送される。また、前記平文ブロックバッファ 8 1 のデータは暗号化し前記暗号文ブロックバッファ 8 2 に転送される。この入出力方法が図 1 8 に示した入出力方法と異なる点は、バッファ長が 1 0 2 4 byte から 2 5 6 byte として示されている点である。

【0 0 6 8】次に、前記共用ファイル暗号プログラム 2 1 に含まれる手動ファイル暗号化ルーチン 2 0 0 0 について説明する。このルーチンは、暗号化を行う平文ファイルのファイル名称と、暗号化後の暗号ファイル名称と宛先リスト 4 1 を入力として動作する。この処理は、図 2 0 に示したファイル暗号化ルーチン 2 0 0 0 と略同じであり、詳細な説明を省略する。

【0 0 6 9】次に、HTML ファイルの暗号化を行う実施例について説明する。この実施例においても、平文の HTML ファイルおよびブラウザでの表示ならびに暗号化した HTML ファイルなどは、実施例 1 の図 2 2 ~ 図 2 6 に示した例と同じであり、その詳細な説明を省略する。

【0 0 7 0】上記の HTML ファイル暗号化を行う HTML ファイル暗号化プログラムの処理フローは実施例 1 の図 2 1 に示した処理フローと同じであり、詳細な説明を省略する。この方法で作成した暗号化 HTML ファイルは図 2 6 に示したものと同じである。以上の暗号化処理を施したファイルは、情報提供者によって前記サーバ情報処理装置の磁気ディスク 5 b に格納される。

【0 0 7 1】次に、前記ブラウザプログラム 1 9 が、W

WWサーバプログラム 2 6 から前記暗号ファイルを受信する際の通信 A P I フックプログラム 2 7 の処理について説明する。まず、受信ファイルデータの入出力とその管理方法から説明する。図 3 3 は、本実施例における受信ファイル復号化の際のデータ入出力を示す図である。

8 3 は、ブラウザプログラム 1 9 が用意する暗号文受信バッファであり、暗号文が含まれている。通信 A P I フックモジュールは、後述する `recv` フック処理の中で、受信した前記暗号文を暗号文ブロックバッファ 8 2 にコピーする。このとき、暗号文ブロックバッファ 8 2 が満たされた場合、該暗号文の復号化処理を実行し平文を平文ブロックバッファ 8 1 にコピーする。さらに、平文をブラウザ 2 3 へ渡すために、前記平文ブロックバッファ 8 1 の中身を前記暗号文受信バッファに上書きコピーする。暗号文受信バッファ 8 3 から暗号文ブロックバッファ 8 2 へコピーした暗号文が、前記暗号文ブロックバッファ 8 2 を満たしていない場合は、復号化処理を行わずに次の受信ファイルデータを待つ。以後、受信した暗号ファイルデータの前記暗号文ブロックバッファ 8 2 へのコピー、前記復号化処理と平文ブロックバッファ 8 1 へのコピー、および暗号文受信バッファ 8 3 への平文コピーを、受信した暗号ファイルデータがなくなるまで繰り返す。

【0 0 7 2】図 3 4 は、オープン中のソケットを管理するためのソケット情報テーブル 9 1 である。ソケット情報テーブル 9 1 は、ソケット記述子 1 9 1 2、受信フラグ 1 9 1 3、受信データポインタ 1 9 1 4、暗号化実効鍵 5 0 9、暗号化宛先リストのサイズ 5 1 0、暗号化宛先リスト 5 1 1、グループ鍵 5 1、実効鍵 6 1、暗号文ブロックバッファポインタ 1 9 1 5、平文ブロックバッファポインタ 1 9 1 6 の各情報を、オープン中の各ソケット毎にポインタ 1 9 0 1 を用いてリスト構造で記憶している。ここでソケット記述子 1 9 1 2 は、ソケットをオープンした際に前記 O S 2 0 から与えられる管理番号である。受信フラグ 1 9 1 3 とは、前記ソケット記述子 1 9 1 2 を使用して暗号ファイルのデータを 1 度も受信していない場合には“ 0 ”、暗号ファイルのデータを 1 回以上受信した場合には“ 1 ”とするフラグである。前記受信データポインタ 1 9 1 4 は、受信したファイルデータから暗号文を読み込む際の起点を示すものであり、後述する `recv` フックルーチン 1 8 1 0 が、データ受信時に受信バッファの先頭アドレスを登録する。暗号文ブロックバッファポインタ 1 9 1 5 と平文ブロックバッファポインタ 1 9 1 6 は、それぞれ図 3 3 における暗号文ブロックバッファ 8 2 と平文ブロックバッファ 8 1 へのポインタであり、復号化処理時にバッファ間で暗号文や平文をコピーする際の起点となるものである。また、前記暗号化実効鍵 5 0 9 と前記暗号化宛先リストサイズ 5 1 0、および暗号化宛先リスト 5 1 1 は、暗号ファイル受信時に前記ヘッダー 5 0 5 から読み込んで登録す

る。なお、該暗号化宛先リスト 5 1 1 は、図 6 に示すようにマスタ鍵によって復号化される。更に、該宛先リスト 4 1 からグループ鍵 5 1 を生成し、また前記暗号化実効鍵 5 0 9 を前記グループ鍵 5 1 で復号化して実効鍵 6 1 を求め、それぞれソケット情報テーブル 9 1 に登録する。

【0 0 7 3】次に、前記通信 A P I フックプログラム 2 7 内の各ルーチンの処理について説明する。図 3 5 は、前記 `send` フックルーチン 1 8 0 0 の処理フローを示す図である。まず、ステップ 1 8 0 1 で、ブラウザプログラム 1 9 が WWW サーバプログラム 2 6 へ送信するデータ中にファイル取得用コマンドである“ G E T ”が存在するか否かを調べる。存在すればステップ 1 8 0 4 で前記ソケット情報テーブル 9 1 を参照し、フックしたソケットの記述子が登録されているか否かを調べる。登録されていれば、ステップ 1 8 0 6 で該当するソケット情報のうち、受信フラグ 1 9 1 3 を“ 0 ”にして、ステップ 1 8 0 7 の `send` 処理を行う。登録されていない場合、ステップ 1 8 0 5 でソケット情報をテーブル 9 0 に登録する。前記ステップ 1 8 0 1 で、送信データ中に前記ファイル取得用コマンドが存在しなければ、フックしたソケットの記述子が前記ソケット情報テーブルに登録されているか否かをステップ 1 8 0 2 で調べる。登録されていれば、ステップ 1 8 0 3 で該当するソケット情報を削除する。登録されていない場合は、ステップ 1 8 0 7 で `send` 処理を実行する。次に、ステップ 1 8 0 8 で `send` 処理のエラーが発生し、かつステップ 1 8 0 9 で該当するソケット情報がテーブルに登録済みの場合は、ステップ 1 8 1 0 で該当するソケット情報を前記ソケット情報テーブル 9 1 から削除し、ステップ 1 8 1 1 で前記ブラウザプログラム 1 9 にリターンする。前記ステップ 1 8 0 8 でエラーが発生しなければ、なにもせずにステップ 1 8 1 1 で前記ブラウザプログラム 1 9 にリターンする。

【0 0 7 4】図 3 6 は、`recv` フックルーチン 1 8 2 0 の処理フローを示す図である。まず、ステップ 1 8 2 1 でブラウザプログラム 1 9 がコールした `recv` の処理を行い、ステップ 1 8 2 2 で前記 `recv` 処理のエラーチェックを行う。このときエラーが発生した場合、ステップ 1 8 3 5 で前記エラーを前記ブラウザプログラム 1 9 にリターンする。なお、エラーに関するメッセージは前記ブラウザプログラム 1 9 が表示するので、`recv` フックルーチン 1 8 1 0 がエラーメッセージを表示する必要はない。エラーなしの場合は、ステップ 1 8 2 3 でソケットの記述子が前記ソケット情報テーブル 9 1 に登録済みであるかどうかを調べる。ここで、`send` フックルーチンの中でソケット情報テーブル 9 1 に登録されたソケット記述子は、WWW サーバ 2 6 が管理する H T M L ファイルやイメージデータファイルの受信に使用するソケットを指す。そこで、ソケット情報テーブル 9

1 に登録済みの場合はステップ 1 8 2 4 で受信データの有無をチェックする。登録されていない場合は、ステップ 1 8 3 5 で受信データをブラウザプログラム 1 9 にリターンする。ステップ 1 8 2 4 で受信データが存在しない場合は、データの受信が完了したことを意味するので、該当するソケット情報を前記ソケット情報テーブル 9 1 から削除し、ステップ 1 8 3 5 で前記ブラウザプログラム 1 9 にリターンする。受信データが存在する場合は、ステップ 1 8 2 5 で `recv` フラグをチェックする。前記 `recv` フラグが " 0 " であれば、前記 `send` 処理 10 に対する 1 回目の受信データであると判断し、ステップ 1 8 2 6 で前記受信データをチェックする。ステップ 1 8 2 7 で、受信データが本発明の暗号化方式により暗号化されたファイルである場合、ステップ 1 8 2 8 でログイン中のユーザーが存在するかチェックする。存在しない場合は、ステップ 1 8 3 8 でメッセージ 1 8 3 7 を表示する。存在する場合は、ステップ 1 8 2 9 で宛先リストに、前記ログインユーザーが含まれているかを調べる。前記ログインユーザーが含まれている場合は、ステップ 1 8 3 0 で前記 `recv` フラグを " 1 " に設定し、ステップ 1 8 3 2 で前記受信ファイルデータを復号化する。一方、前記宛先リストに前記ログインユーザーが含まれていない場合は、ステップ 1 8 3 1 でメッセージ 1 8 3 6 を表示し、ステップ 1 8 3 4 でソケット情報を前記ソケット情報テーブル 9 1 から削除してリターンする。一方、前記ステップ 1 8 2 7 で、受信ファイルデータが暗号化されていない場合は、ステップ 1 8 3 4 でソケット情報を前記ソケット情報テーブルから削除してリターンする。

【0075】図 3 7 は、`closesocket` フック 30 ルーチン 1 8 4 0 の処理フローを示す図である。まずステップ 1 8 4 1 では、前記ブラウザプログラム 1 9 がコールした `closesocket` 関数の引数に指定されたソケット記述子が、ソケット情報テーブル 9 1 に登録されているかを調べる。登録されている場合は、ステップ 1 8 4 2 で前記ソケット記述子に対応するソケット情報をテーブル 9 0 から削除し、ステップ 1 8 4 3 の `closesocket` 処理を実行して前記ブラウザプログラム 1 9 にリターンする（ステップ 1 8 4 4）。一方、前記ソケット情報テーブルに登録されていない場合は、 40 ステップ 1 8 4 3 の `closesocket` 処理を実行してリターンする。

【0076】以上説明した通信 API フックプログラム 2 7 の各ルーチンの処理によって、所望のページの復号化をブラウザプログラム 1 9 からは透過に実現することができる。

【0077】次に、本実施例において、実際の WWW 運用システムの一例について図 3 8 を用いて説明する。1 4 0 は、代理サーバ情報処理装置であり、代理サーバプログラム 1 8 により、前記クライアント情報処理装置 1 50

1 0 と前記サーバ情報処理装置 1 2 0 との間の送受信データを中継する。また、前記代理サーバプログラム 1 8 は、前記サーバ情報処理装置 1 2 0 が前記クライアント情報処理装置 1 1 0 からの要求に応じて発信した HTML ファイルやイメージデータファイルを、ディスクコントローラ 4 c を介して磁気ディスク 5 c 内にキャッシュする機能を持つ。以後、前記クライアント情報処理装置 1 1 0 からファイル転送要求を出した場合、該ファイルが前記磁気ディスク 5 c 内にキャッシュされていれば、前記代理サーバプログラム 1 8 が該キャッシュファイルを前記クライアント情報処理装置 1 1 0 へ転送する。本実施例においては、機密情報に関するファイルは全て暗号化されて前記サーバ情報処理装置 1 2 0 の磁気ディスク 5 b に格納されている。したがって、ファイル転送時に前記磁気ディスク 5 c にキャッシュされる機密情報は全て暗号ファイルとなる。

【0078】

【発明の効果】以上説明したように、本発明はファイルの暗号化において、復号化を許可するユーザーのリストをファイルに付加するグループ暗号用いるので、複数のユーザーが同一のファイル利用する場合にも暗号化ファイルが一個で済み且つ安全であるという効果がある。

【0079】また、暗号化の対象ディレクトリと復号を許可するユーザーの宛先リストをユーザー毎に設定可能であるので、複数のユーザーがディレクトリを共用し該ディレクトリ内のファイルを異なる宛先リストを用いてファイル暗号化を行うことができるという効果がある。

【0080】また、本発明は、アプリケーションプログラムが解釈し処理する文法に従って記述された平文ファイルのデータ暗号化において、前記文法における前記アプリケーションプログラム処理時に処理対象外として無視する注釈文とするための記述子を前記暗号化後のデータに付加し暗号ファイルとして格納するので、本暗号化システムに対応した復号プログラムを具備しない情報処理装置で前記暗号化ファイルを読み出した場合にも、異常動作することがないという効果がある。

【0081】また、本発明は、アプリケーションプログラムから OS に発行されるファイル操作要求をフックするフック手段を設け、前記ファイル操作要求発行時に自動的に暗号化あるいは復号化を行うので、ユーザーが暗号化を忘れることがないという効果がある。

【0082】また、本発明は、WWW サーバが予め暗号化されたファイルを管理するので、該 WWW サーバプログラムを具備するサーバ情報処理装置に直接アクセスしても機密情報が漏れないという効果がある。また、本発明は、WWW サーバが予め暗号化されたファイルを管理するので、該ファイル転送時に代理サーバによってキャッシュされる機密情報も暗号ファイルとなるため、代理サーバ情報処理装置に直接アクセスしても機密情報が漏れないという効果がある。

【0083】また、本発明は、ブラウザプログラムから通信モジュールに発行される通信API呼び出しをフックするフック手段を設け、前記データ受信API呼び出し時に自動的に復号化を行うので、ユーザーにとって復号化のための操作が不要であるという効果がある。

【図面の簡単な説明】

【図1】本発明の実施例における共用ファイル暗号システムの一構成例を示す図。

【図2】本発明の実施例におけるID情報29の内容の一例を示す図。

【図3】本発明の実施例において、宛先リストのハッシュ値を計算し前記グループ鍵51を作成する手順の一例を示す図。

【図4】本発明の一実施例におけるICカード130内の前記グループ暗号プログラム28の処理内容を示すフロー図。

【図5】本発明の一実施例におけるファイルの暗号化手順と暗号化後のファイルの構成を示す図。

【図6】本発明の一実施例におけるファイルの復号化手順を示す図。

【図7】本発明の一実施例におけるユーザーから見た論理的なディレクトリの構成を示す図。

【図8】本発明の一実施例におけるファイルの自動暗号化、復号化を導入した後の物理的なディレクトリ構成を示す図。

【図9】本発明の一実施例における宛先リストファイル901、前記暗号ファイル情報ファイル902の内容の一例を示す図。

【図10】本発明の一実施例におけるICカード130へのログイン処理ルーチン1000の処理フローを示す図。

【図11】本発明の一実施例におけるICカード130からのログアウト処理ルーチン1100の処理フローを示す図。

【図12】本発明の一実施例におけるファイルオープンフックルーチン1200の処理フローを示す図。

【図13】本発明の一実施例におけるファイルクリエイトフックルーチン1300の処理フローを示す図。

【図14】本発明の一実施例におけるリードフックルーチン1400の処理フローを示す図。

【図15】本発明の一実施例におけるライトフックルーチン1500の処理フローを示す図。

【図16】本発明の一実施例におけるクローズフックルーチン1600の処理フローを示す図。

【図17】本発明の一実施例におけるファイルリネームフックルーチン1700の処理フローを示す図。

【図18】本発明の一実施例におけるファイル暗号化・復号化時のデータの入出力方法を示す図。

【図19】本発明の一実施例におけるオープン中の暗号ファイルを管理するためのファイルハンドルテーブル9

0の構成を示す図。

【図20】本発明の一実施例における手動ファイル暗号化ルーチン2000の処理フローの一例を示す図。

【図21】本発明の第一の実施例における第一のサーバ情報処理装置120のディレクトリ構造を示す図。

【図22】本発明の一実施例における平文のHTMLファイルの一例を示す図。

【図23】本発明の一実施例において平文のHTMLファイルをブラウザで表示した画面の例を示す図。

10 【図24】本発明の一実施例における暗号化したHTMLファイルの一例を示す図。

【図25】本発明の一実施例において暗号化した平文のHTMLファイルをブラウザで表示した画面の例を示す図。

【図26】本発明の一実施例における暗号化したHTMLファイルの他の例を示す図。

【図27】本発明の第一の実施例における前記共用ファイル暗号プログラム21と前記ファイルI/Oフックプログラム22を構成するプログラムルーチンを示す図。

20 【図28】本発明の実施例における暗号ファイル受信システムの一構成例を示す図。

【図29】本発明の実施例における前記共用ファイル暗号プログラム21と前記通信APIフックプログラム27を構成するプログラムルーチンを示す図。

【図30】本発明の一実施例におけるファイルの暗号化手順と暗号化後のファイルの構成を示す図。

【図31】本発明の一実施例におけるファイルの復号化手順を示す図。

30 【図32】本発明の一実施例におけるファイル暗号化・復号化時のデータの入出力方法を示す図。

【図33】本発明の一実施例における受信ファイル復号化時のデータの入出力方法を示す図。

【図34】本発明の一実施例におけるオープン中のソケット情報を管理するためのソケット情報テーブル91の構成を示す図。

【図35】本発明の一実施例におけるsendフックルーチン1800の処理フローを示す図。

【図36】本発明の一実施例におけるrecvフックルーチン1820の処理フローを示す図。

40 【図37】本発明の一実施例におけるclosesocketフックルーチン1840の処理フローを示す図。

【図38】本発明の実施例において実際のWWWを運用するシステムの一構成例を示す図。

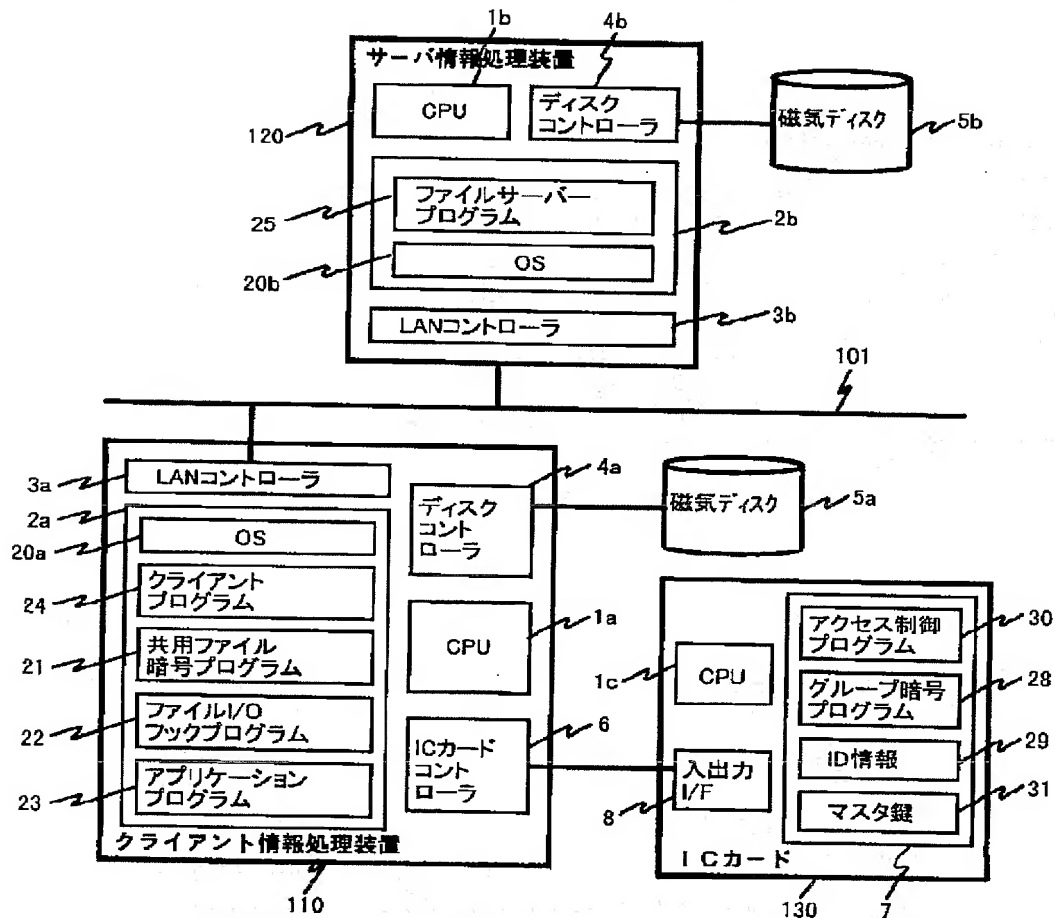
【符号の説明】

- 1 CPU
- 2 メモリ
- 3 LANコントローラ
- 4 ディスクコントローラ
- 5 磁気ディスク
- 50 6 ICカードコントローラ

7 不揮発メモリ  
 8 入出力インタフェース  
 19 ブラウザプログラム  
 18 代理サーバプログラム  
 20 OS  
 21 共用ファイル暗号プログラム  
 22 ファイルI/Oフックプログラム  
 23 アプリケーションプログラム  
 24 クライアントプログラム  
 25 ファイルサーバプログラム  
 26 WWWサーバプログラム

27 通信APIフックプログラム  
 28 グループ暗号プログラム  
 29 ID情報  
 30 アクセス制御プログラム  
 31 マスタ鍵  
 32 ハッシュ関数  
 101 LAN  
 110 クライアント情報処理装置  
 120 サーバ情報処理装置  
 130 ICカード  
 140 代理サーバ情報処理装置

【図1】

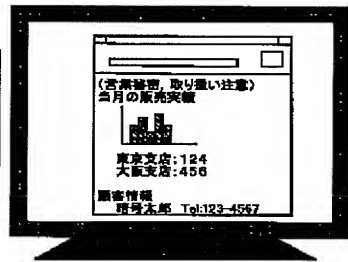


【図 2】

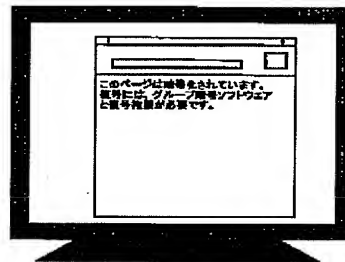
ID 情報

カテゴリ	データ	コード
1 生年月日	19900101	-
2 氏 名	復号太郎	-
3 性 別	男性	1
4 氏名コード	677991234	-
5 事業所	システム研究所	676
6 部	第四部	04
7 課	安全課	402
8 職名	主任	5
9 分野	セキュリティ	X5
10 出身地	神奈川	24

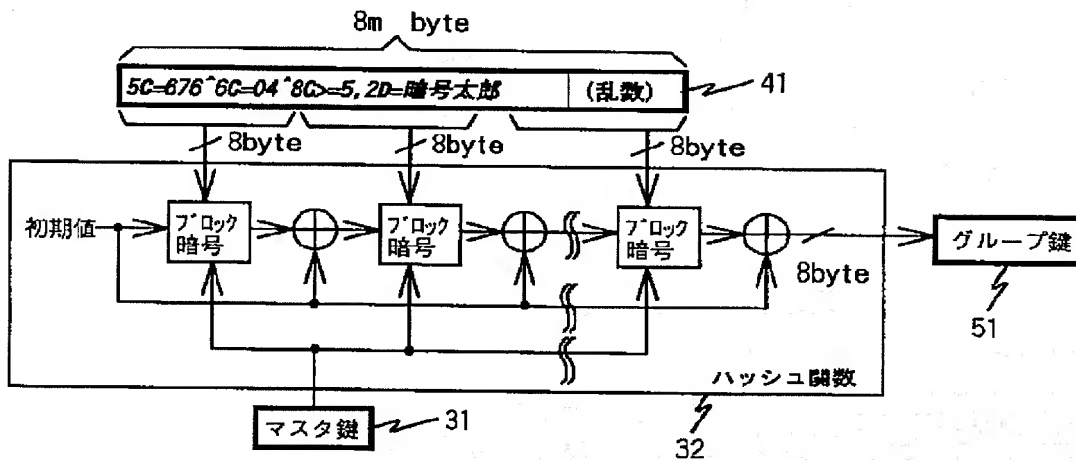
【図 2 3】



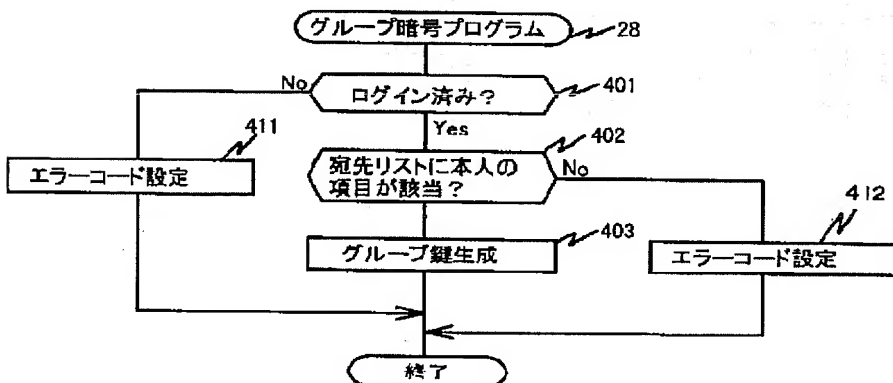
【図 2 5】



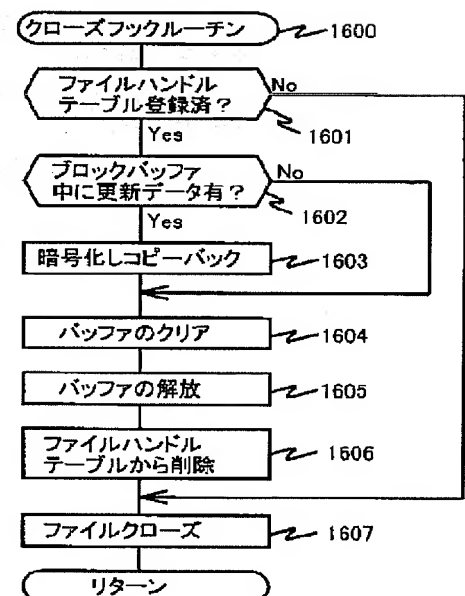
【図 3】



【図 4】

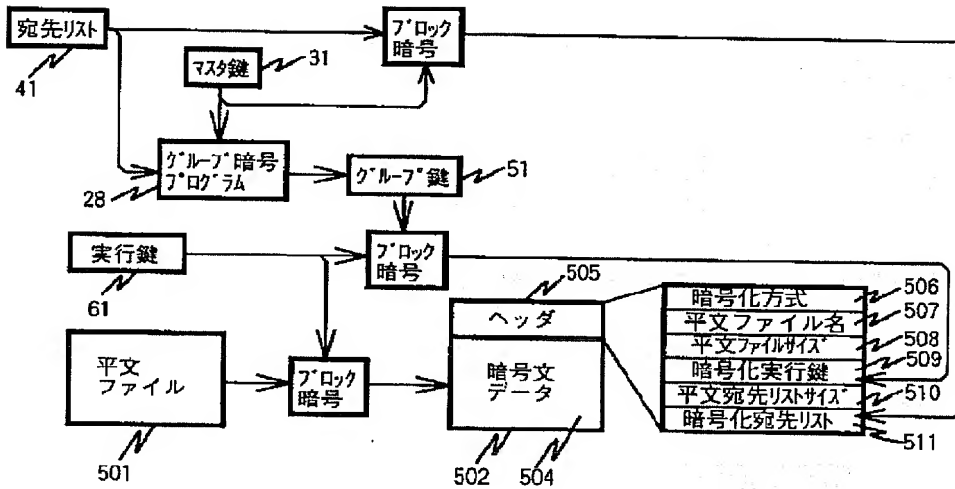


【図 1 6】

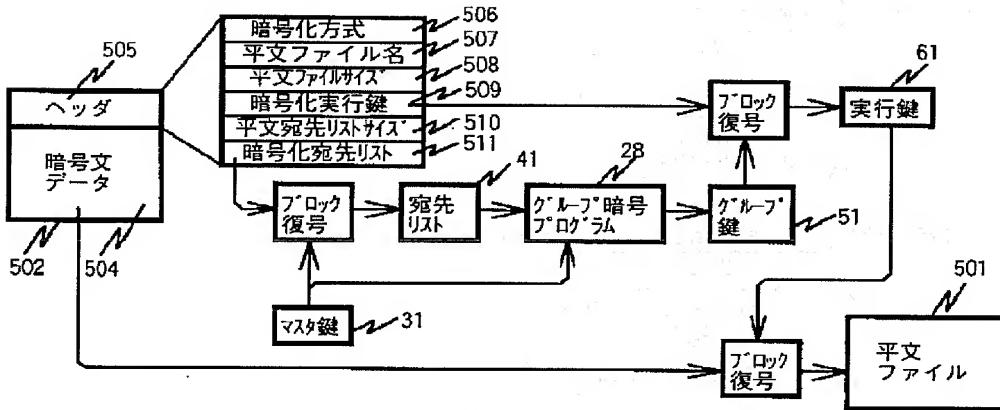




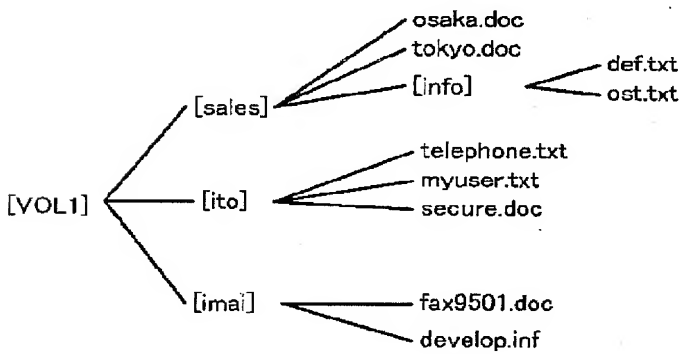
【図 5】



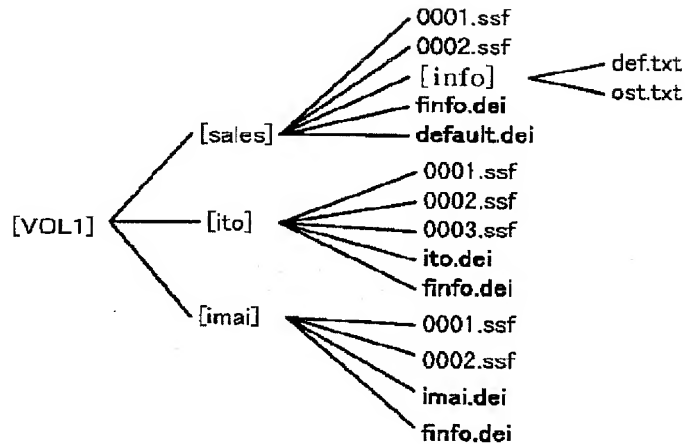
【図 6】



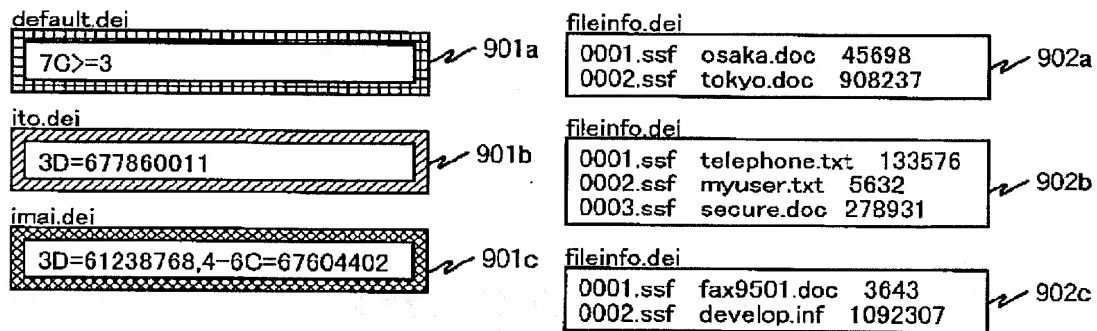
【図 7】



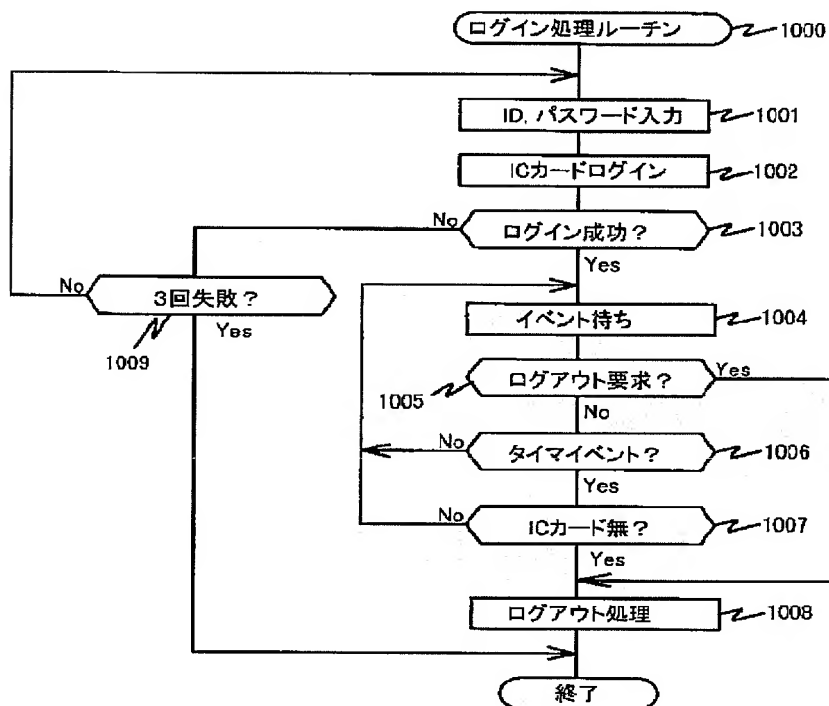
【图 8】



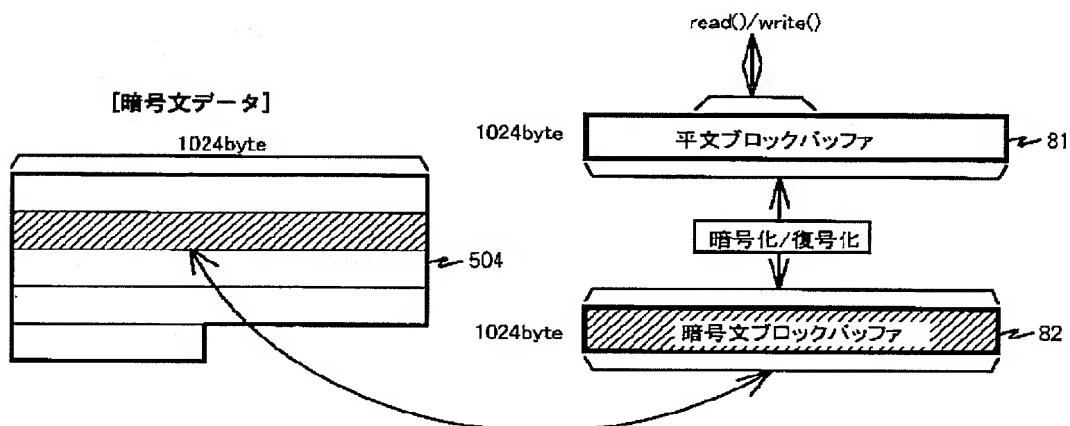
【図 9】



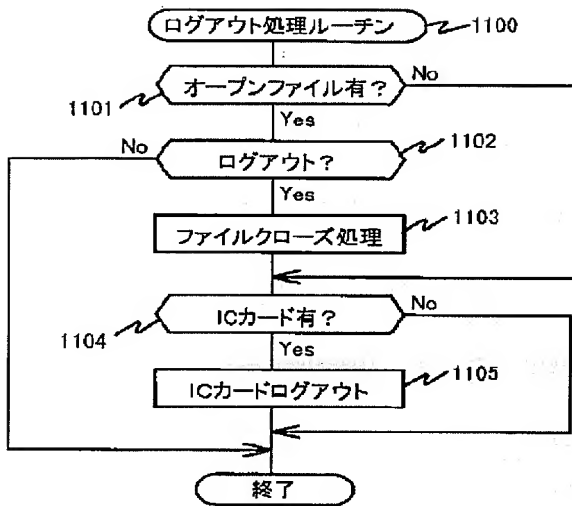
【図 10】



【図 18】



【図11】



【図22】

secret.htm

```

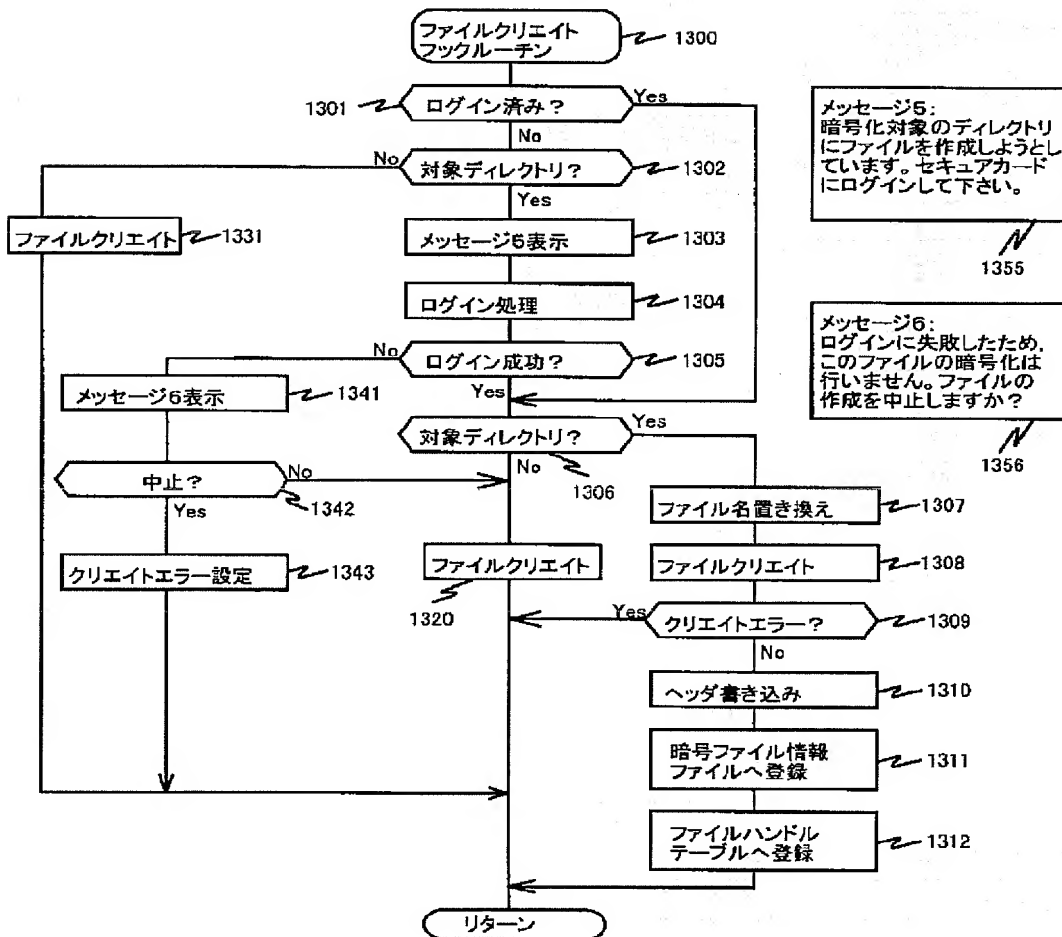
<HTML>
<HEAD><TITLE>Sales information</TITLE></HEAD>
<BODY>
<IMG SRC="image/graph.gif"><P>
(営業秘密, 取り扱い注意)
当月の販売実績
  東京支店: 124
  大阪支店: 456

顧客情報
  暗号太郎 Tel:123-4567

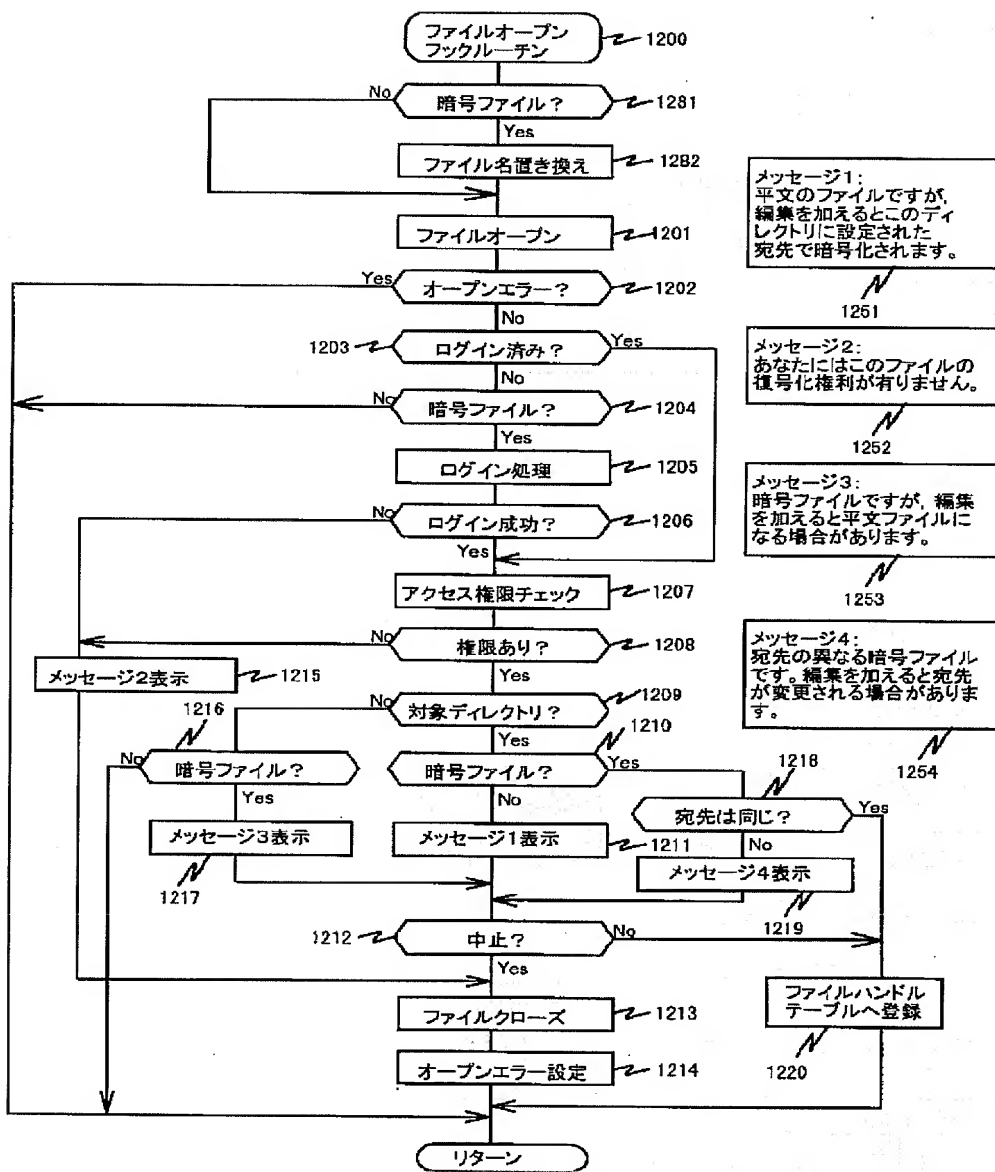
</BODY>
</HTML>
  
```

2201

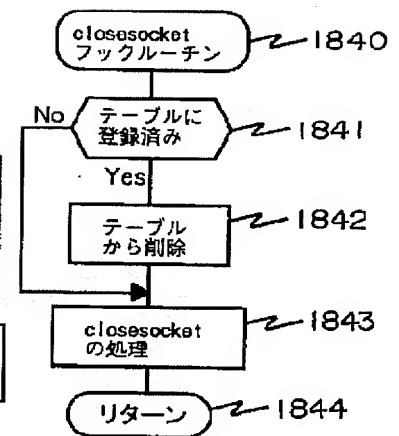
【図13】



【図 1 2】



【図 3 7】

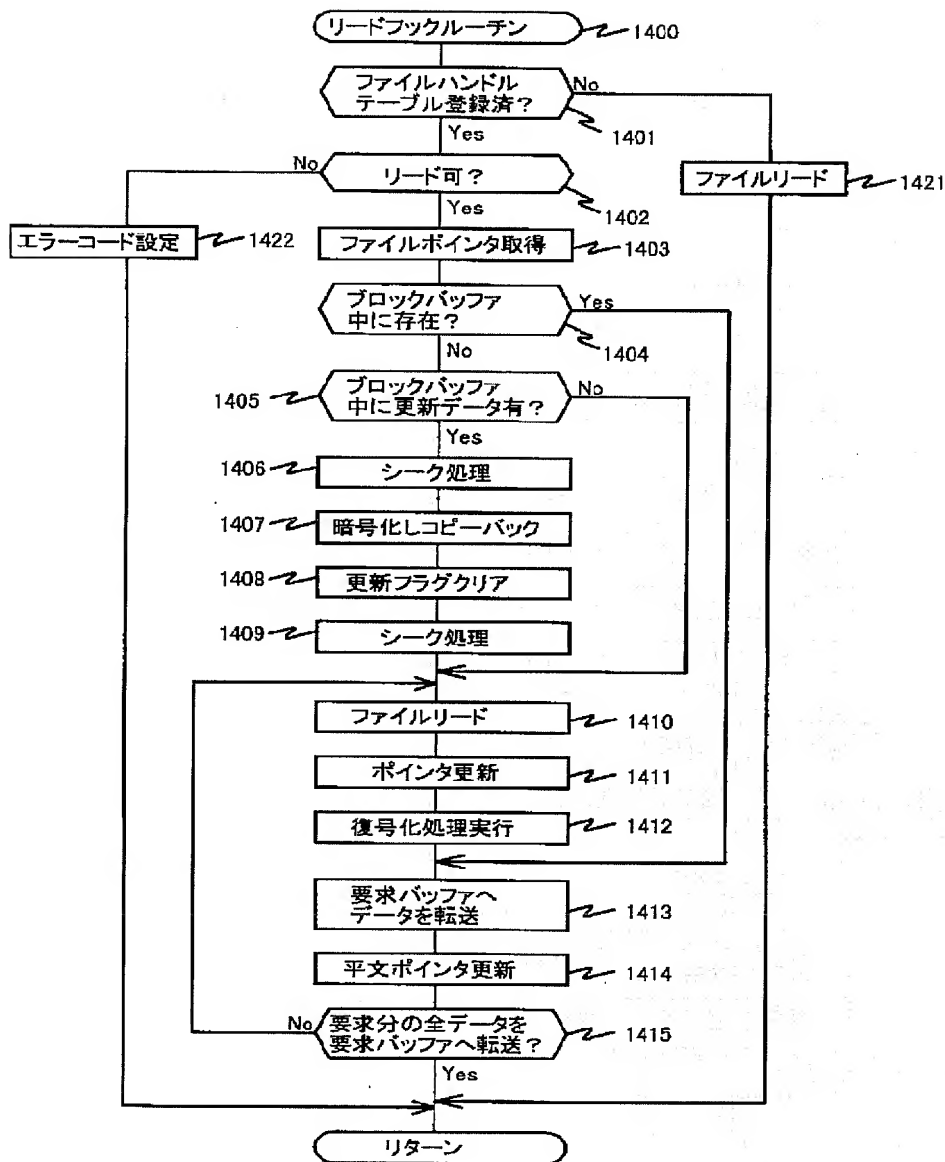


【図 1 9】

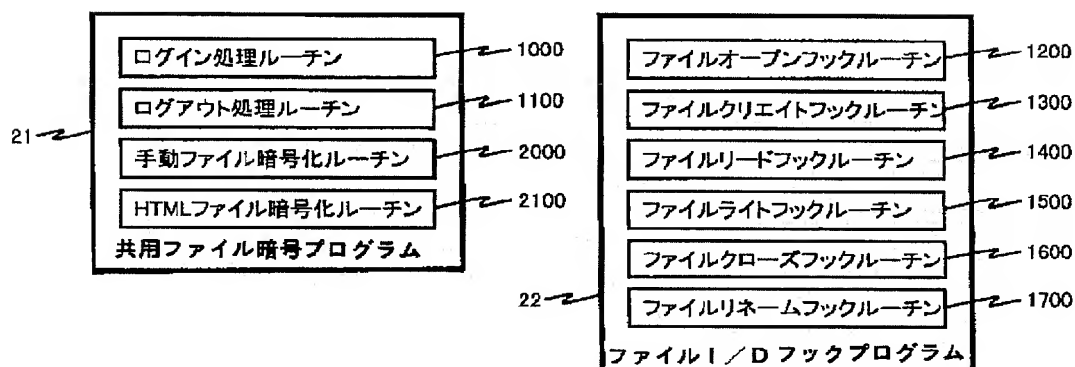
ファイルハンドルテーブル

ポインタ	ポインタ	ポインタ	90
ハンドル番号	ハンドル番号	ハンドル番号	1901
オープンモード	オープンモード	オープンモード	1902
平文ファイルポインタ	平文ファイルポインタ	平文ファイルポインタ	1903
暗号ファイルポインタ	暗号ファイルポインタ	暗号ファイルポインタ	1904
バッファポインタ	バッファポインタ	バッファポインタ	1905
更新フラグ	更新フラグ	更新フラグ	1906
宛先リスト	宛先リスト	宛先リスト	1907
			41

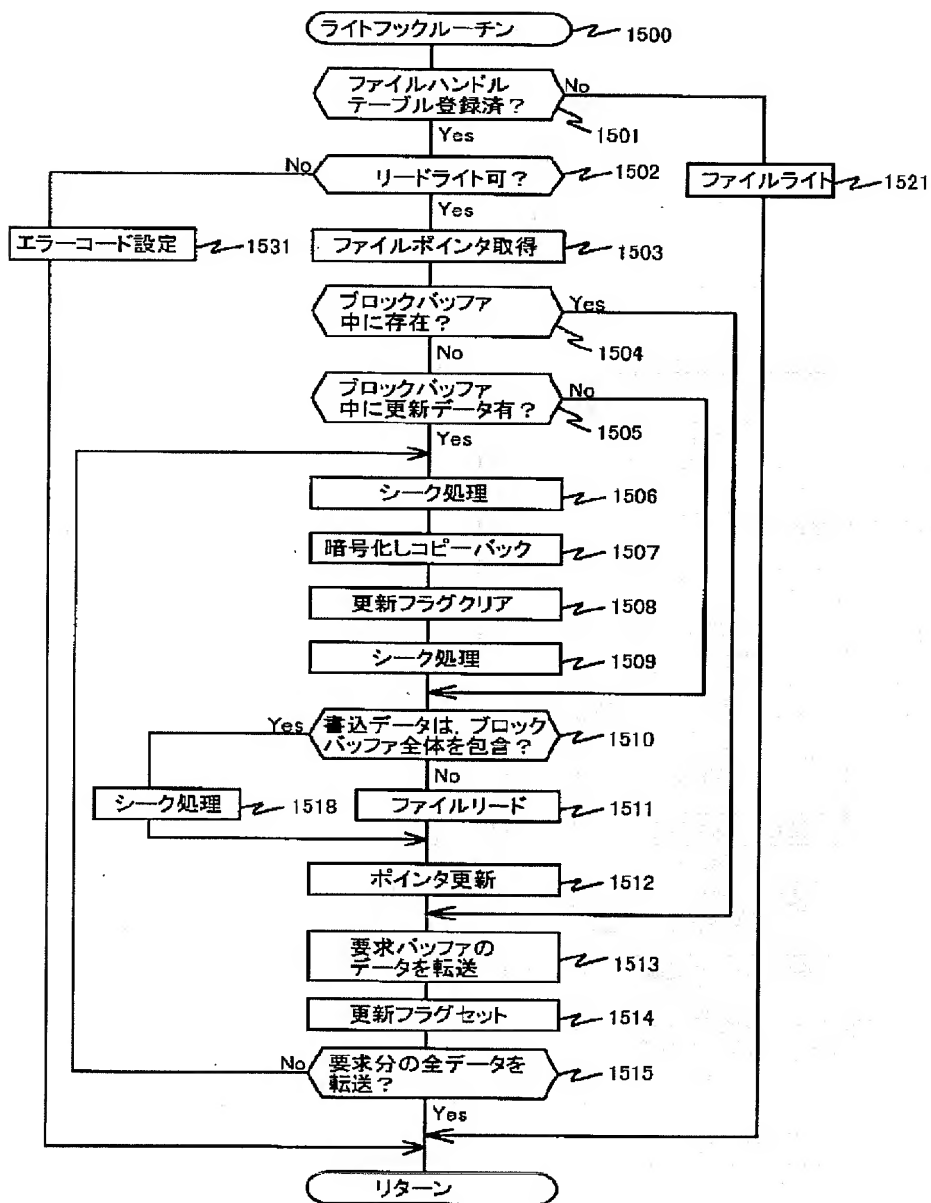
【図 1 4】



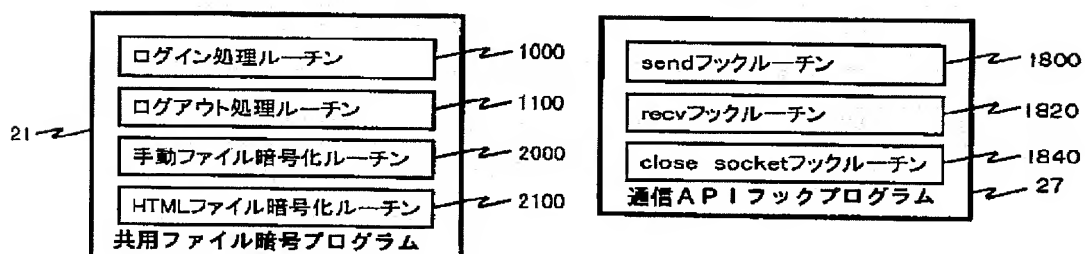
【図 2 7】



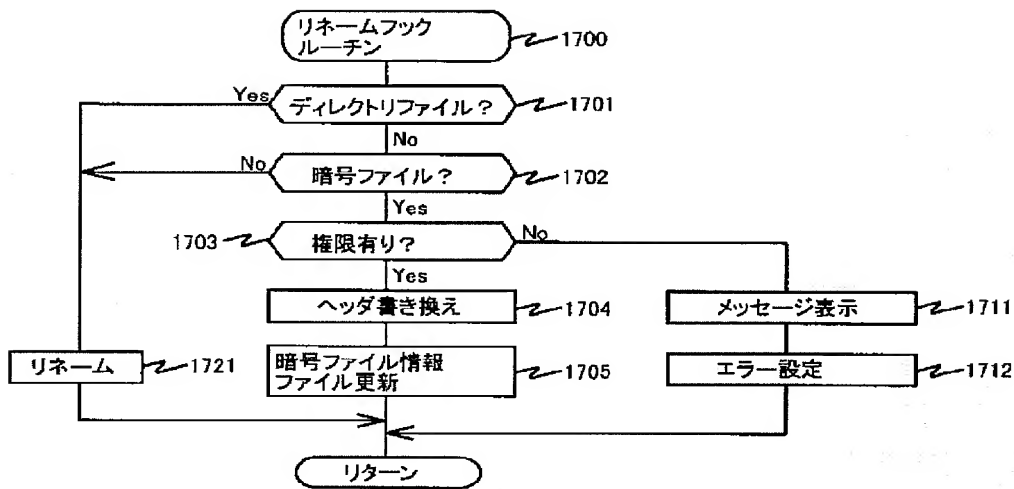
【図 1 5】



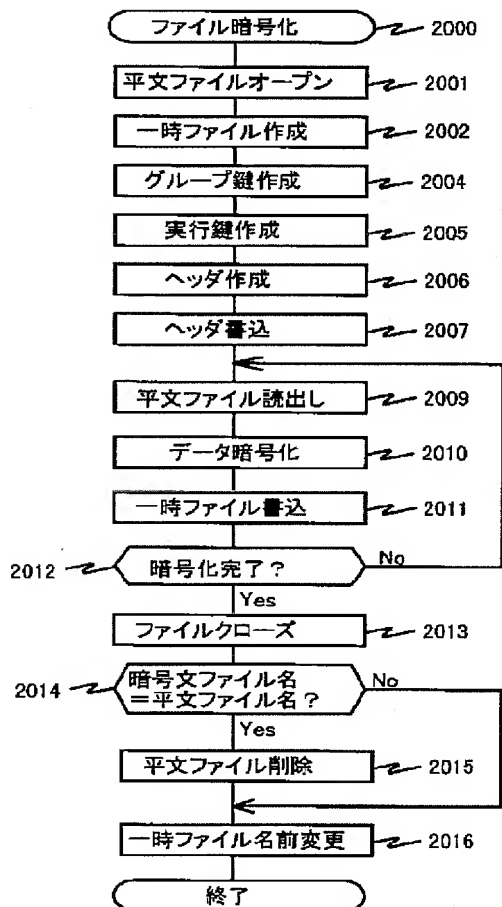
【図 2 9】



【図17】



【図20】



【図24】

secret.htm

```

<HTML>
<HEAD><TITLE>Sales information</TITLE></HEAD>
<BODY>
このページは、暗号化されています。
復号には、グループ暗号ソフトウェアと復号権限が必要です。
<!--GroupEncryption V1.0JKasdhfDsLWHewjkdU
ILAYHq3eu3'(UIyuak.f4n48hjBf1se.a489basdJK
LSHdskdakSJDnds;3wKJDSdKosidja()IOsd2heHWQ
whewqhOQW+EH;o.ahue.hweo89seSJDLA+JD9uiJda
klu23eklau09hhjss;lqwen24 vkljdallana/\;ak
jD)DIj#L4h3nq-->
</BODY>
</HTML>
  
```

```

<HTML>
<HEAD><TITLE>Sales information</TITLE></HEAD>
<BODY>
このページは、暗号化されています。
復号には、グループ暗号ソフトウェアと復号権限が必要です。
<!--
  
```

```

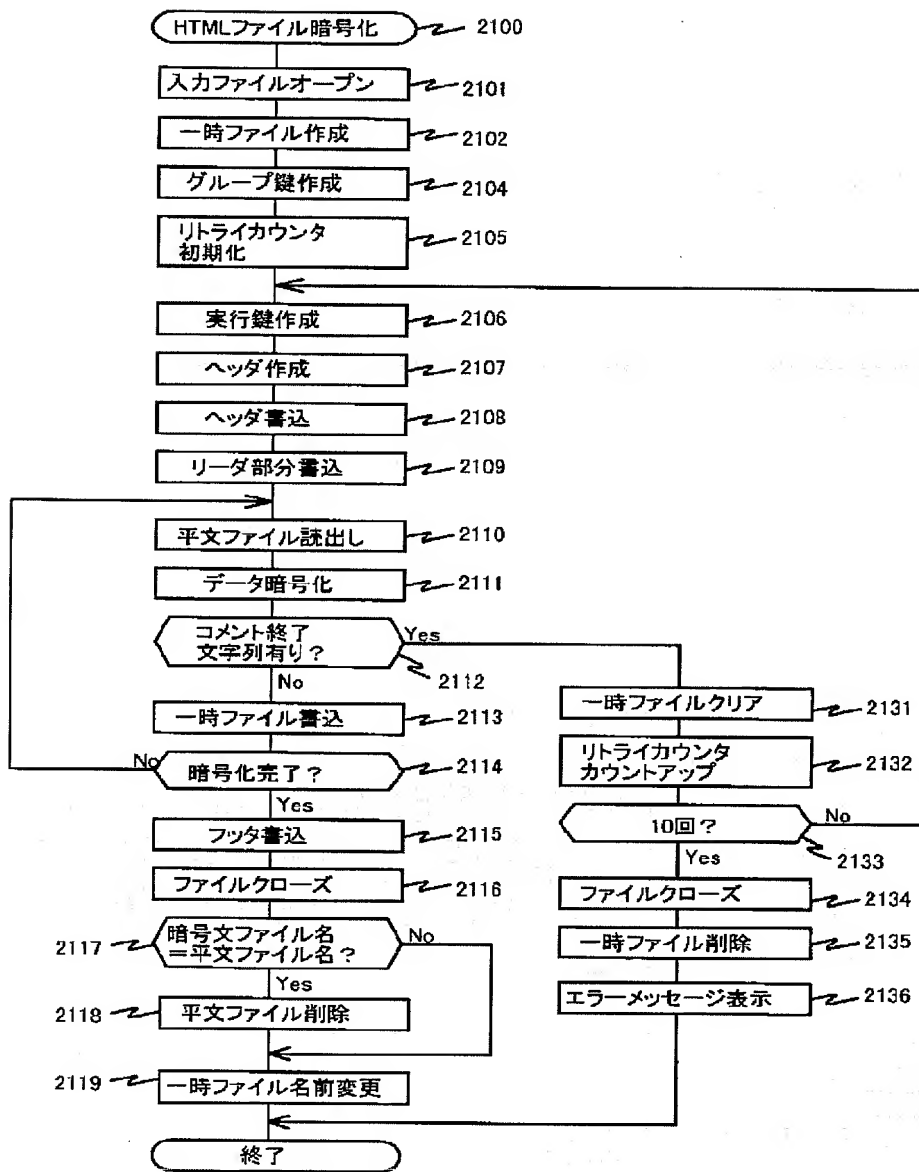
GroupEncryption V1.0JKasdhfDsLWHewjkdU
ILAYHq3eu3'(UIyuak.f4n48hjBf1se.a489basdJK
LSHdskdakSJDnds;3wKJDSdKosidja()IOsd2heHWQ
whewqhOQW+EH;o.ahue.hweo89seSJDLA+JD9uiJda
klu23eklau09hhjss;lqwen24 vkljdallana/\;ak
jD)DIj#L4h3nq
  
```

```

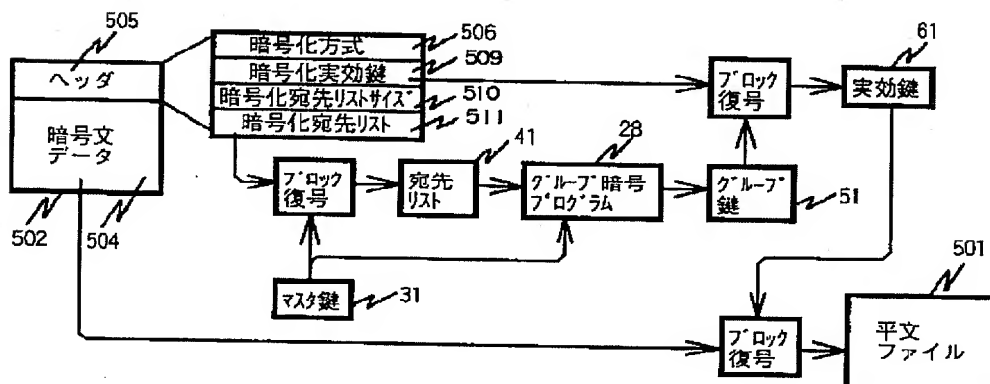
-->
</BODY>
</HTML>
  
```



【図 2 1】



【図 3 1】



【図 2 6】

secret.htm

```

<HTML>
<HEAD><TITLE>Sales information</TITLE></HEAD>
<BODY>
このページは、暗号化されています。
復号には、グループ暗号ソフトウェアと復号権限が必要です。

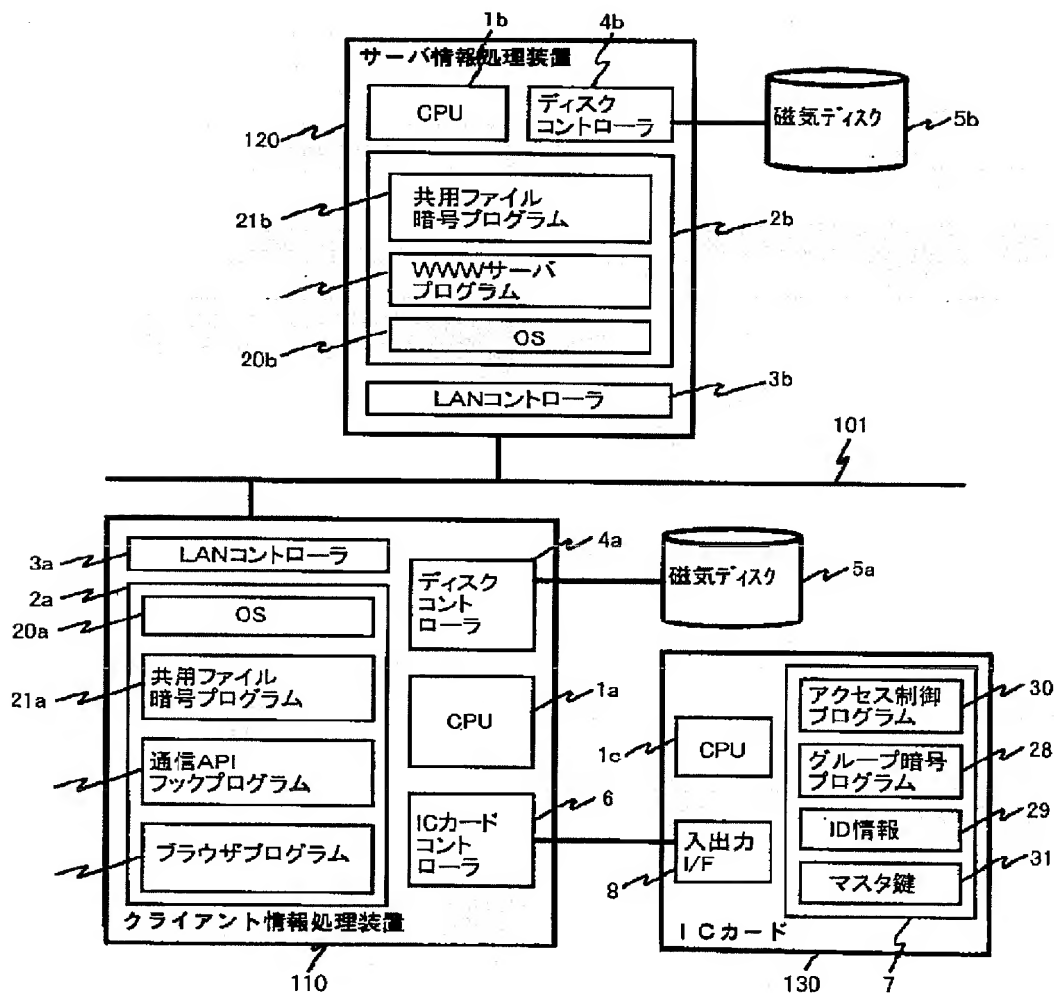
<!--GroupEncryption V1.04A4B61736448664473
4C574865776A6B6455494C4159487133657533270D
0A2855497975616B2E66346E3438686A42666C7365
2E61343839626173644A4B4C534844736B64616B53
4A446E64733B33774B4A440D0A53444B6F7369646A
612829494F73643268654857517768657771684F51
572B45483B6F2E616875652E6877656F383973650D
0A534A444C412B4A443975694A64616B6C75323365
6B6C6175303968686A73733B6C7177656E32342076
6B6C6A64616C6C616E612F5C3B0D0A616B6A442944
496A234C3468336E71-->

</BODY>
</HTML>

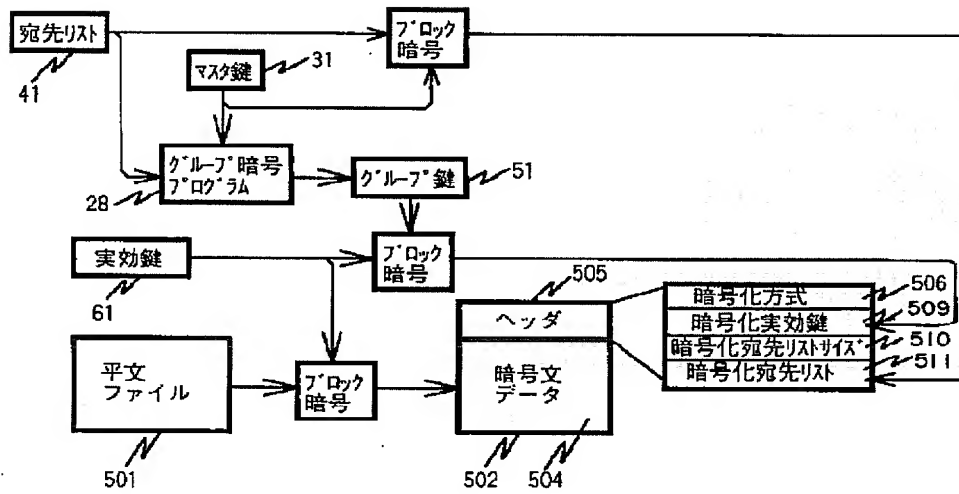
```

2601

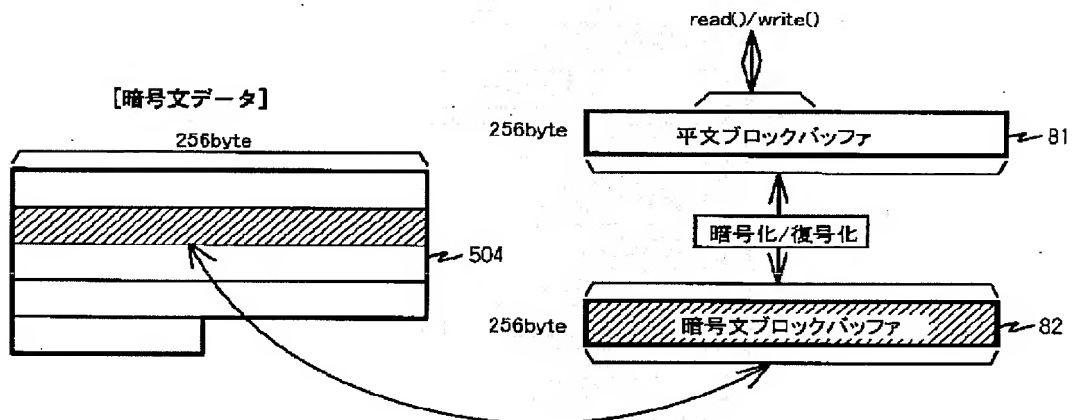
【図 2 8】



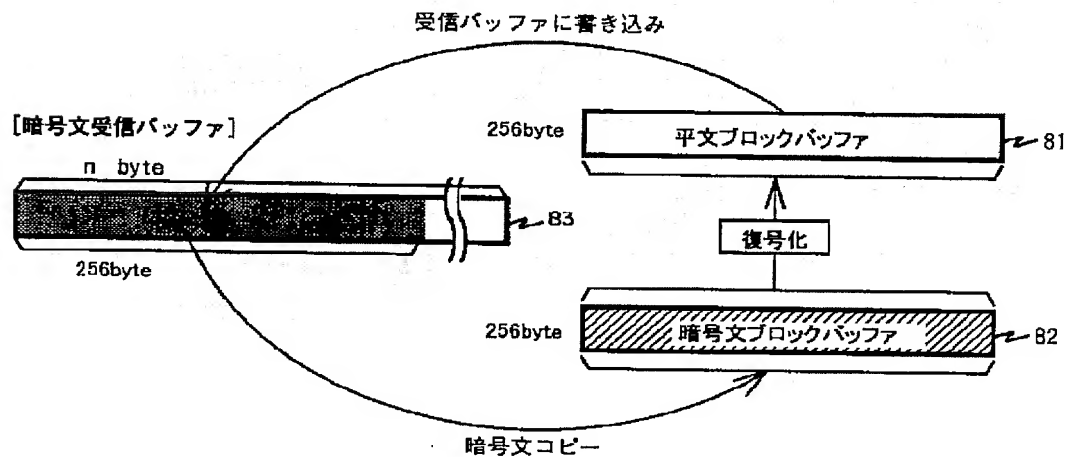
【図 3 0】



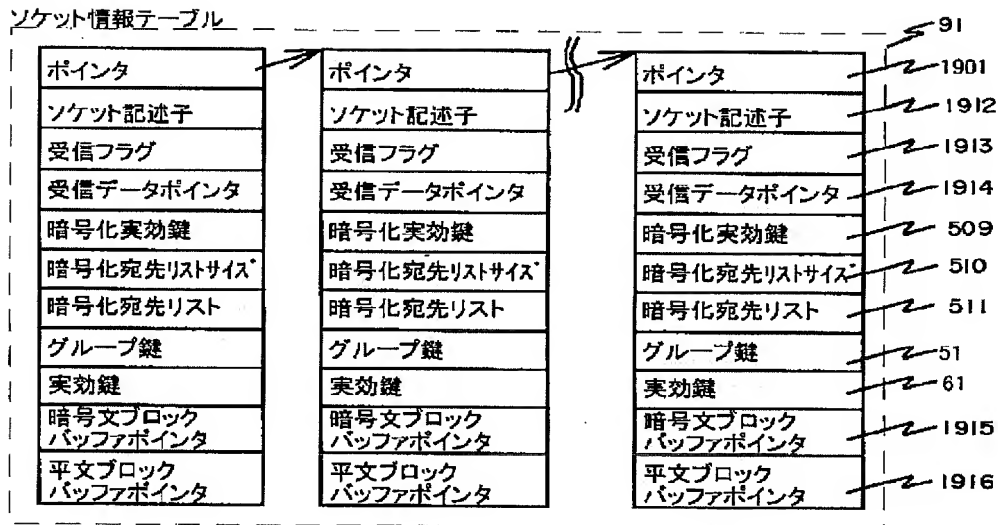
【図 3 2】



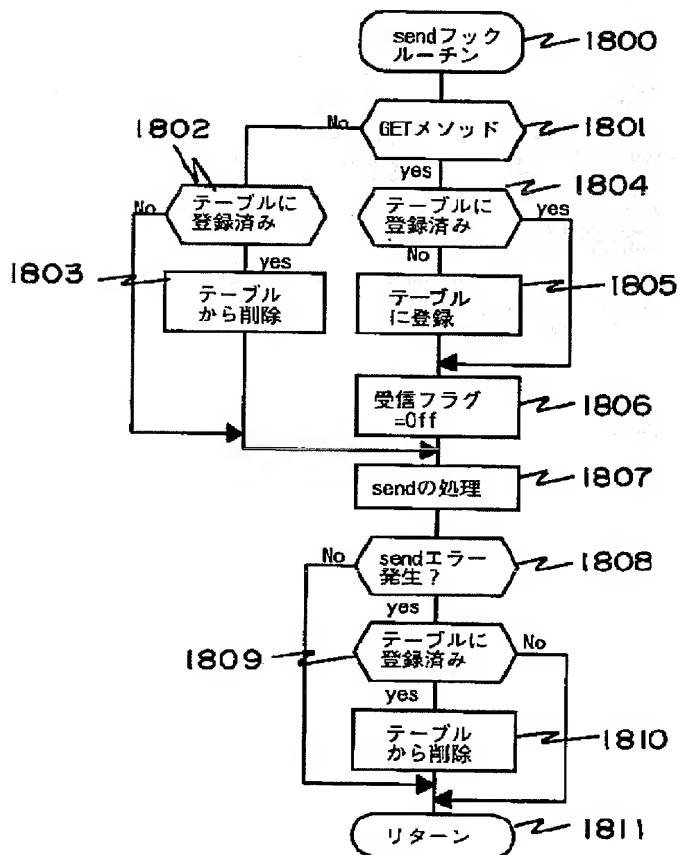
【図 3 3】



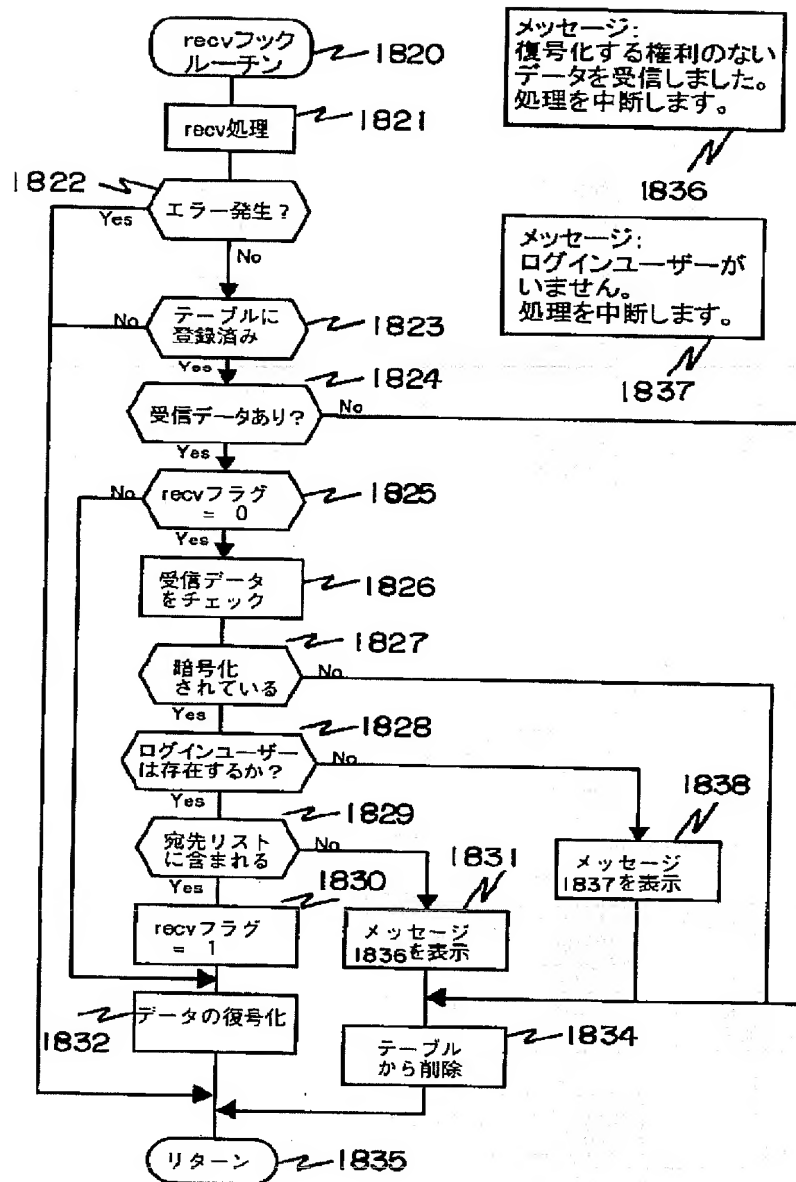
【図34】



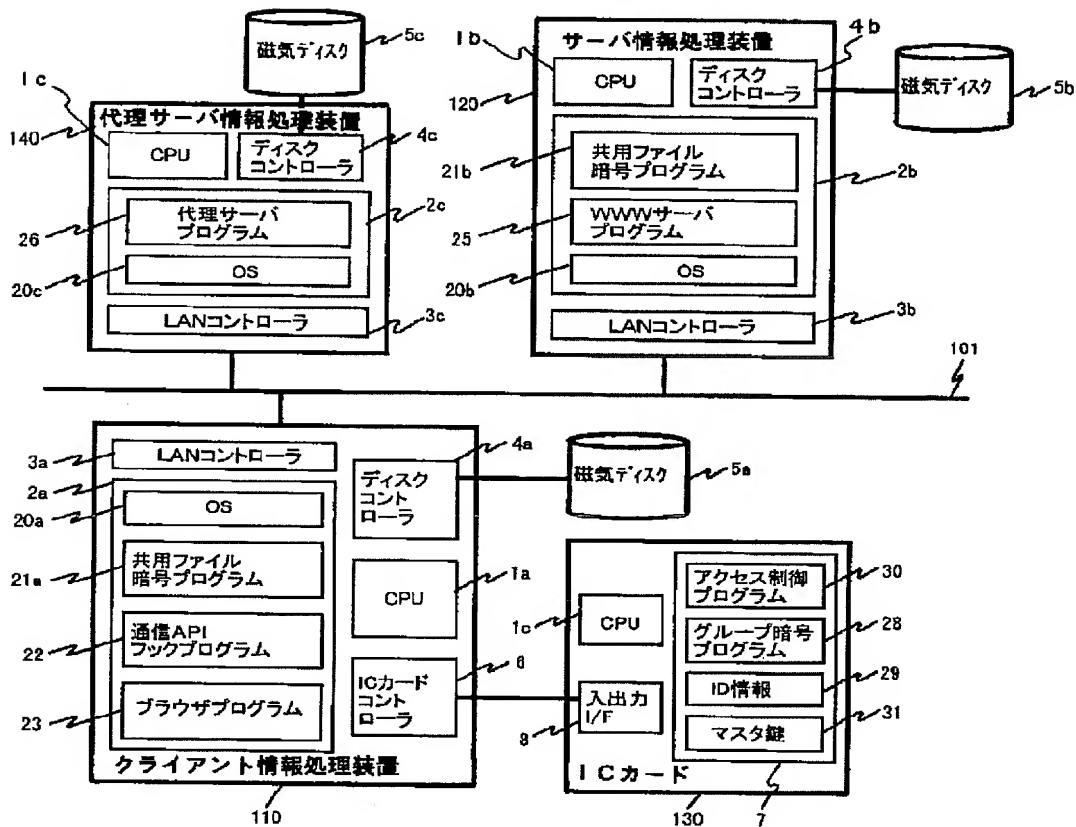
【図35】



【図36】



【図 3 8】



フロントページの続き

(72) 発明者 梅木 久志  
 神奈川県川崎市麻生区王禅寺1099番地 株  
 式会社日立製作所システム開発研究所内  
 (72) 発明者 大津 豊  
 神奈川県横浜市戸塚区戸塚町5030番地 株  
 式会社日立製作所ソフトウェア開発本部内

(72) 発明者 森藤 元  
 神奈川県川崎市麻生区王禅寺1099番地 株  
 式会社日立製作所システム開発研究所内  
 (72) 発明者 清水 麻由子  
 神奈川県横浜市戸塚区戸塚町5030番地 株  
 式会社日立製作所ソフトウェア開発本部内

